NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

Anode Sheath Contributions in Plasma Thrusters

by

John Forrest Riggs

March, 1994

Thesis Advisor: Oscar Biblarz

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
March 1994

3. REPORT TYPE AND DATES COVERED
Engineer's and Master's Thesis

4. TITLE AND SUBTITLE ANODE SHEATH CONTRIBUTIONS IN PLASMA THRUSTERS

5. FUNDING NUMBERS

6. AUTHOR(S) Riggs, John Forrest

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE
A

13. ABSTRACT *(maximum 200 words)*

Contributions of the anode to Magnetoplasmadynamic (MPD) thruster performance are considered. High energy losses at this electrode, surface erosion, and sheath/ionization effects must be controlled in designs of practical interest. Current constriction or spotting at the anode, evolving into localized surface damage and considerable throat erosion, is shown to be related to the electron temperature's (T_e) rise above the gas temperature (T_o). An elementary one-dimensional description of a collisional sheath which highlights the role of T_e is presented. Computations to model the one-dimensional sheath are attempted using a set of five coupled first-order, nonlinear differential equations describing the electric field, as well as the species current and number densities. For a large temperature nonequilibrium (i.e., $T_e >> T_o$), the one-dimensional approach fails to give reasonable answers and a multidimensional description is deemed necessary. Thus, anode spotting may be precipitated by the elevation of T_e among other factors. A review of transpiration cooling as a means of recouping some anode power is included. Active anode cooling via transpiration cooling would result in (1) quenching T_e, (2) adding "hot" propellant to exhaust, and (3) reducing the local electron Hall parameter. However, significant technical problems remain.

15. NUMBER OF PAGES 106

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

Approved for public release; distribution is unlimited.

Anode Sheath Contributions in Plasma Thrusters

by
John F. Riggs
Lieutenant Commander, United States Navy
B.A., University of Kansas, 1982

Submitted in partial fulfillment
of the requirements for the degrees of

AERONAUTICAL & ASTRONAUTICAL ENGINEER
and
MASTER OF SCIENCE IN ASTRONAUTICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
March 1994

Author: John F. Riggs
John F. Riggs

Approved by: Oscar Biblarz
Oscar Biblarz, Thesis Advisor

Fred Schwirzke
Fred Schwirzke, Second Reader

D. J. Collins
D.J. Collins, Chairman
Department of Aeronautical Engineering

Richard S. Elster
Richard S. Elster, Dean of Instruction

ABSTRACT

Contributions of the anode to Magnetoplasmadynamic (MPD) thruster performance are considered. High energy losses at this electrode, surface erosion, and sheath/ionization effects must be controlled in designs of practical interest. Current constriction or spotting at the anode, evolving into localized surface damage and considerable throat erosion, is shown to be related to the electron temperature's (T_e) rise above the gas temperature (T_o). An elementary one-dimensional description of a collisional sheath which highlights the role of T_e is presented. Computations to model the one-dimensional sheath are attempted using a set of five coupled first-order, nonlinear differential equations describing the electric field, as well as the species current and number densities. For a large temperature nonequilibrium (i.e.,$T_e >> T_o$), the one-dimensional approach fails to give reasonable answers and a multidimensional description is deemed necesssary. Thus, anode spotting may be precipitated by the elevation of T_e among other factors. A review of transpiration cooling as a means of recouping some anode power is included. Active anode cooling via transpiration cooling would result in (1) quenching T_e, (2) adding "hot" propellant to exhaust, and (3) reducing the local electron Hall parameter. However, significant technical problems remain.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENT

I dedicate this work to my dear wife Lin, for her support and understanding during my absence as a geographical bachelor of almost three years.

My sincere thanks to my parents for encouraging me to pursue my ambitions from childhood on, and to Oscar Biblarz, Ph.D., for patiently helping me to understand plasma physics.

I. INTRODUCTION

Several types of space flight propulsion systems have been developed over the years. These include chemical, nuclear, electric and solar propulsion. The majority of space thrusters to date have been chemical rockets. Although the Chinese used rockets over 800 years ago, true development of rocket propulsion took place during this century [Ref. 1]. Chemical thrusters give high thrust-to-weight ratios, larger than unity, and have been fully developed in the form of space launch vehicles and attitude control thrusters. In contrast, other propulsion systems have been developed only to the proof-of-concept stage, and essentially remain at this stage of development. Nuclear propulsion was studied with the NERVA (Nuclear Engine for Rocket Vehicle Application) thruster in the 1960's, and abandoned [Ref. 2:pp. 517-519]. Electric propulsion flights during the 1960's included the U.S. SERT-1 (Space Electric Rocket Test) in 1964 and the U.S.S.R. Yantari-1 rocket in 1966. Solar-electric propulsion was demonstrated via the SERT-2 rocket in 1970, powering the electric thruster from power generated by solar cells. Further electric propulsion research flights in the 1980's included the U.S. Navy's NOVA-1 satellite in 1981, and Japan's MS-T4 satellite,launched from the Space Shuttle. Beyond this, nonchemical thrusters have only been used in auxiliary roles, such as station-keeping and attitude

control on geosynchronous satellites. NASA's Project PATHFINDER in the mid-1980's proposed the use of a megawatt-level electric plasma thruster for a manned Mars mission. However, development of this project was never funded.

In comparing the different propulsion schemes, a primary performance indicator is specific impulse, defined as the ratio of thrust to the rate of propellant usage, or alternately, propellant effective exhaust velocity (u_e), divided by the sea-level gravitational constant, (g_o).

$$I_{sp} = \frac{\dot{m} u_e}{\dot{m} g_o} = \frac{u_e}{g_o} \quad \text{sec} \tag{1}$$

Chemical rockets are inherently limited in performance by the total energy available in the fuel/oxidizer combustion process, so that the total enthalpy available for conversion into exhaust kinetic energy is limited. Exhaust velocity is also limited by material heating limitations of the combustion chamber and nozzle throat, and "frozen flow Losses" (unrecoverable energy deposition in internal modes of the gas) [Ref. 3:pp. 4-5]. Peak specific impulse for liquid chemical propellants is presently on the order of 450 seconds. This capability is completely sufficient for the tasks of launch to low earth orbit (LEO), attitude control, station keeping, and such missions. However, for missions such as manned interplanetary exploration, chemical propulsion can be shown to be clearly inadequate. A comparative analysis of a Mars

mission using chemical and electric propulsion systems shows the large difference in mass payload ratio (final mass/initial launch mass) for the two systems. A chemical system using a high impulse Hohman ellipse trajectory delivers a maximum of approximately 10% to 18% of the launch mass to the Martian surface [Ref. 4:p. 115]. In comparison, an electric system using a low impulse spiral trajectory could deliver from 20% to 60% of the launch mass, depending on the desired transit time. Each mission assumes transit from low Earth orbit to Mars orbit. An electric propulsion system would still need a high thrust propulsion system to reach the Martian surface Ref. 5:pp. 344-346]. The large difference in payload ratio is due to the much larger exhaust velocity and more efficient use of fuel by electric propulsion. Thus, some form of electric or hybrid electric thruster would seem to be in order for such interplanetary missions. However, due to the low thrust-to-weight ratio of electric thrusters, they must be launched into orbit by other means. Their usefulness is restricted to space thrusters, not to launch systems.

With specific impulses of as high as 10,000 seconds, electric propulsion offers the performance envelope needed for manned interplanetary missions. Electric propulsion is divided into three types of thrusters: electrostatic, electrothermal, and electromagnetic. The type relevant to this work is the magnetoplasmadynamic (MPD) thruster, an electromagnetic propulsion system that utilizes the Lorentz force created by an electric current together with its induced magnetic field to propel a gas that has been heated to the plasma state. According to electromagnetic theory, a conductor carrying a current produces an induced magnetic force perpendicular to

the current. The applied electric field and its induced magnetic field interact to produce the Lorentz force ($\vec{F} = \vec{J}\times\vec{B}$) perpendicular to both fields on the conductors. This briefly summarizes the concept behind the "self-field" MPD accelerator [Ref. 2:pp. 485-486]. MPD performance is enhanced by adding magnetic coils to the thruster, thus strengthening the magnetic field and, as a consequence, the Lorentz force and thrust. This thruster is appropriately called an "applied-field" MPD thruster. MPD thrusters have shown specific impulses of up to 7,000 seconds and efficiencies as high as 70% [Ref. 6:pp. 2-3]. Performance of MPD thrusters is limited by several factors, including electrode erosion, current spotting, frozen flow losses, and electrode power deposition. Specifically, anode power deposition is the single largest power loss mechanism in MPD thrusters operating at submegawatt power levels [Ref. 7]. In the following work, we review and analytically model the MPD anode, including the sheath and anode potential drop.

II. LITERATURE REVIEW

Anode losses significantly limit magnetoplasmadynamic (MPD) thruster performance. Much effort has been placed on characterizing these losses and on the nature of power deposition in the anode [Refs. 8-14]. As much as 80% of thruster total power may end up being deposited in the anode at sub-megawatt power levels [Refs. 8,15]. This power deposition together with current constriction at the anode surface present serious problems to thruster cooling and performance, as well as to anode lifetime. Before any practical design can be achieved, a more thorough understanding of the phenomena at the anode, particularly the anode sheath, must be gained. Studies have shown that the anode power fraction depends on thruster power, current, mass flow rate, and the parameter $J^2/\dot{m}$ [Refs. 8,12,13,16]. It has also been shown that the anode fall voltage is inversely proportional to anode current density [Refs. 13,16]. It is believed that a better understanding of the role of an elevated electron temperature, of current flow dimensionality, and of current unsteadiness are prerequisites for the evolution of any practical MPD thruster design.

Computer codes that accurately describe observed data from steady-state MPD thrusters have been developed [Refs. 17-19]. However, these codes do not adequately describe observed data from quasi-steady thruster experiments. It has been suggested that the lack of proper electrode modelling (i.e., sheaths and fall potentials) in these

codes may explain this discrepancy [Ref. 6]. Limited analytical work has been done in modelling the sheath and ambipolar regions at the anode, influenced perhaps by the difficult set of coupled, nonlinear partial differential equations involved. Hugel [Ref. 12] and Subramaniam [Ref. 20] address the influence of the sheath region, but do not model the electric field, temperature, or sheath fall voltage.

Given the minuscule extent of the sheath versus thruster anode curvature, the problem at first appears one-dimensional in nature. A one-dimensional, collisional, equilibrium solution can satisfactorily reproduce the observed electric field and charge density distributions for the entire sheath and ambipolar regions for a sheath where the electron temperature equals that of the heavy species [Ref. 21]. However, this model cannot describe any decrease in current density away from the surface, or current constriction, at the anode surface which might be necessary in nonequilibrium. A two-dimensional model, developed by Biblarz and Dolson [Ref. 14], represents these phenomena and predicts the voltage drop in the region. It is shown that the sheath must account for a majority of the anode voltage drop, and that the sheath extent must be greater than the Debye length [Refs. 14,21]. Thus, a combination of one- and two-dimensional approaches appears to better describe sheath behavior. Incorporation of modelling of this sort may improve the ability of the computer codes cited above to properly describe quasi-steady thrusters.

Next, a description of the anode region is presented in order to delineate some of the possible effects of temperature.

III. ANODE DESCRIPTION

A. THRUSTER GEOMETRY DESCRIPTION

The majority of plasma thrusters to date have consisted of a central cathode rod surrounded by an annular shell anode, as shown in Figure 1 [Ref. 23]. The thruster illustrated is sufficient to produce needed thrust at current levels above one kiloamp. Below this level, an external magnetic field produced by an annular magnet is needed to ensure sufficient Lorentz force on the plasma propellant to meet thrust requirements. [Ref. 8]. As illustrated in Figure 1, the $\vec{j}\times\vec{B}$ body force simplifies into an axial $(j_r B_\theta)$ body force, which provides direct electromagnetic thrust ("blowing"), and a radial $(-j_z B_\theta)$ body force, which provides electromagnetic compression of the plasma and a subsequent pressure force along the cathode surface ("pumping"). [Ref. 6]

A notable exception to this geometry is the Stationary Plasma Thruster (SPT), a design from the former Soviet Union. The SPT is an example of a plasma propulsion system known as a Hall Current Plasma accelerator. An electric field is applied axially to a stream of flowing plasma, in addition to a magnetic field with a strong radial component, which is applied by an external electromagnet. When the axial electric field is applied and a current flows through the plasma, an azimuthal component of current is induced, i.e., the "Hall" current.

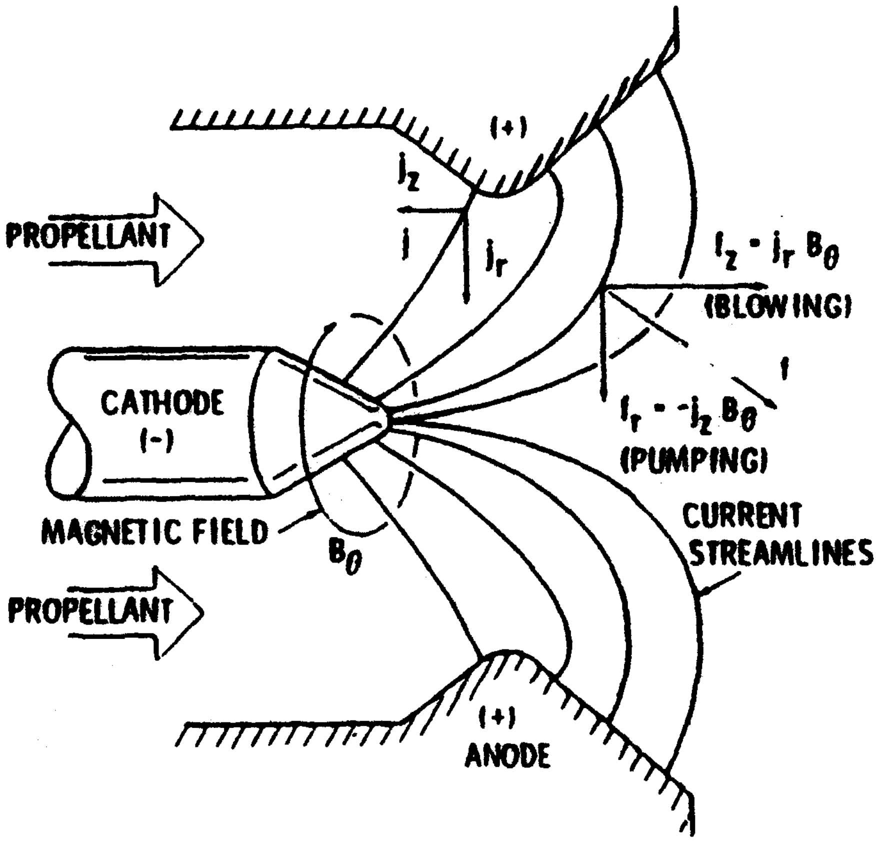

Figure 1 - Magnetoplasmadynamic (MPD) Thruster, with Axial and Radial Forces on Plasma Indicated. [Ref. 23]

Thrust is produced by electrostatically accelerating the ions in the plasma, as well as through the induced Lorentz force mentioned above. A strong radial magnetic field is applied to the plasma, whose properties are controlled to make the electron Larmor radius[1] small compared to the mean free path[2], while the ion Larmor radius is comparatively large. As a consequence, electron mobility in the axial direction is greatly reduced. Thus, the electric field energy is given mainly to the ions, p[illegible]ng axial ion acceleration. Collisions with neutral particles serve to accelerate the entire neutral plasma. [Ref. 24]

A pair of the final prototype design developed, the SPT-100, have been acquired by NASA recently from Fakel Enterprise in Kaliningrad, Russia, and are undergoing performance evaluation at the Jet Propulsion Laboratory. Designed at the Kurchatov Institute of Atomic Energy (IAE) in Moscow, USSR in the 1960's, smaller versions of the SPT-100 (SPT-50 and SPT-70) were flown beginning in 1972[3]. A specific impulse of 1,600 seconds and 50% efficiency, as well as space flights of fifty similar thrusters is claimed. The specific operational characteristics of the thruster are not well understood presently. Bohm diffusion of electrons and a phenomenon called "near-wall conductivity" have been proposed to explain the thruster's operation. This thruster is shown in Figure 2. [Ref. 25]

---

[1] Larmor radius is the radius of the helix traversed by a charged particle moving in a magnetic field.

[2] Mean free path is the distance traveled by a particle before making a collision.

[3] The suffix (i.e.,"-70") indicates the characteristic diameter of the thruster in millimeters.

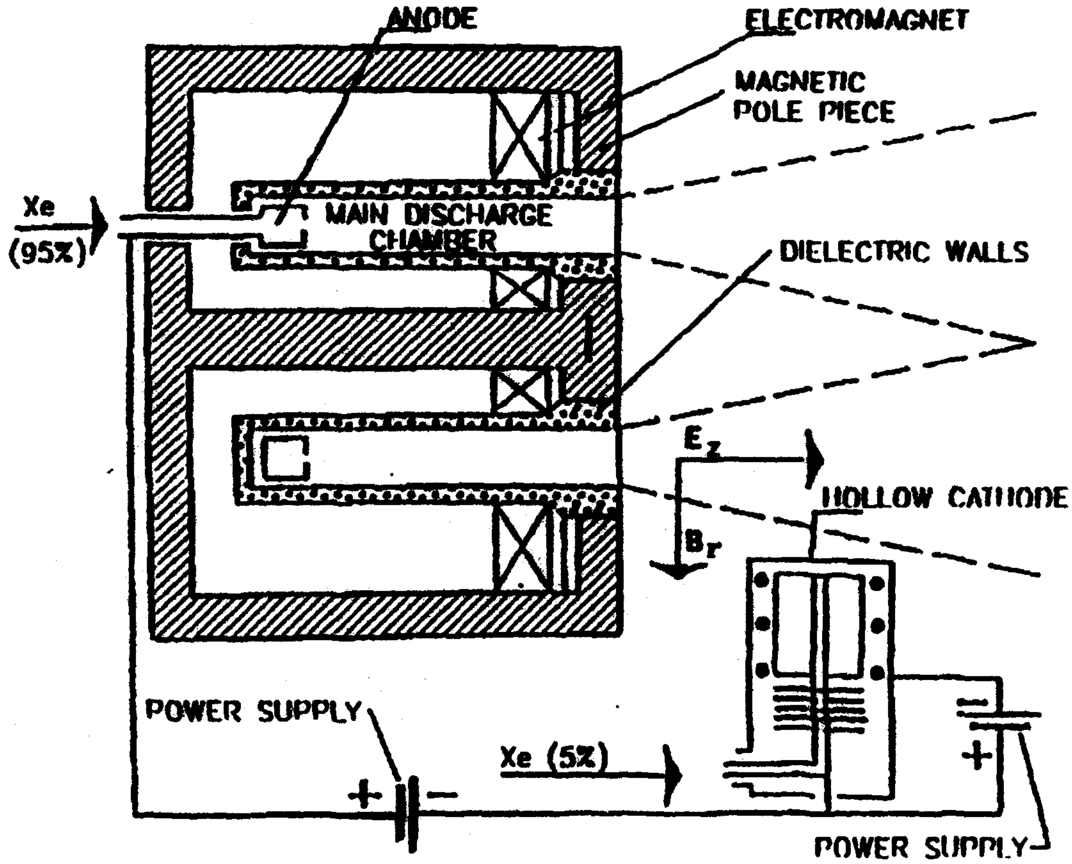

Figure 2 - Stationary Plasma Thruster [Ref. 25]

B. ELEMENTARY SHEATH FORMULAE DESCRIPTION

1. Discussion

Voltage losses and anode power deposition account for most of the inefficiency of plasma thrusters. In order to understand these losses, the anode region must be understood and related phenomena explained and modelled. As shown in Figure 3, a substantial drop in voltage occurs in a short distance from the anode surface.

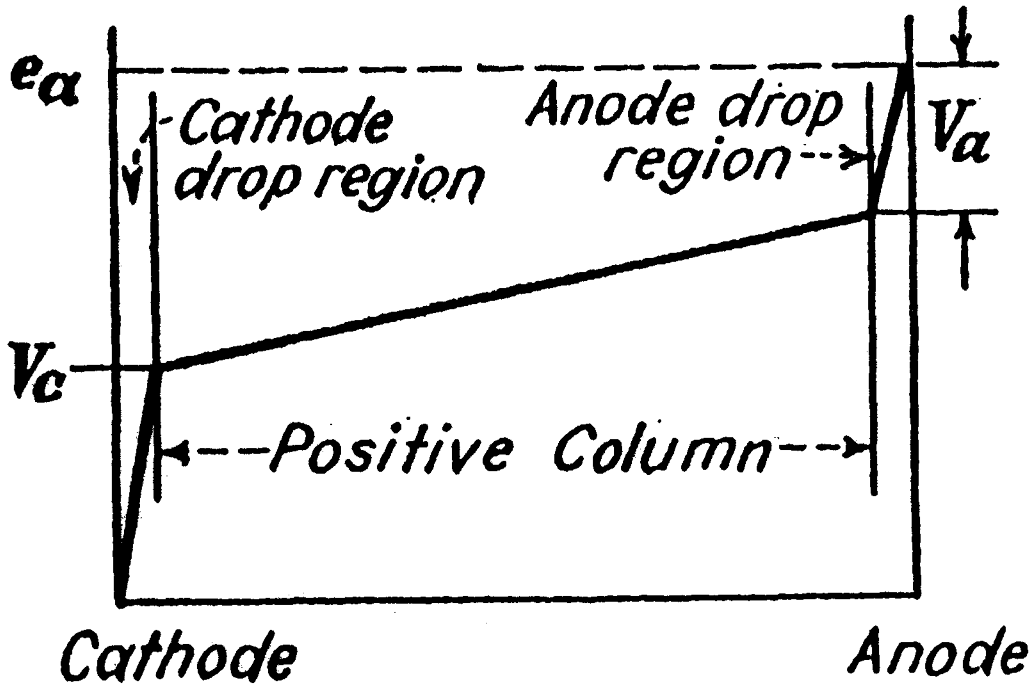

Figure 3 - Electric Field Between Two Electrodes, Including "Positive Column". [Ref. 27]

The anode fall region may be divided into two parts, the sheath and ambipolar regions. The plasma attempts to adjust itself near electrodes so as to shield the main body of the plasma from the electric field [Ref. 26]. The sheath is the region closest

to the anode surface within which the ion and electron number densities are unequal, with the electrons dominating the region. A high electron space charge exists at the anode surface. This is caused by the anode collecting incoming electrons in completing the arc current with the cathode. Positive ions are produced within the sheath by electron impact of neutral gas molecules, and the ions are repelled toward the cathode. At the cathode end of the anode drop region, the density of positive ions is high enough to almost neutralize the electron space charge, thus forming the positive column or core plasma. The essential positive ion current is created in this way near the anode. A more complete description of this process may be found in Cobine [Ref. 27] and von Engel [Ref. 28]. A fundamental characteristic of plasma behavior is its tendency toward electrical neutrality. Whenever <u>local</u> charge concentrations arise or external potentials are introduced into a system, these are shielded out in a distance known as the "Debye length"[*]. This distance must be much smaller than the system dimension for the ionized gas to be considered a plasma [Ref. 29]. Equation (2) gives the Debye length [Ref. 26].

$$\lambda_d = \sqrt{\frac{e_o kT}{n_\infty e^2}} = 69.0 \sqrt{\frac{T}{n_\infty}} \quad (m) \tag{2}$$

[*]The Debye length effectively describes the radius of a shell around a charged particle outside of which the potential of the particle is not seen.

Another distance of interest is the electron mean free path, or distance traveled by a particle before making a collision. Equation (3) is from a derivation of Lin, Resler, and Kantrowitz [Refs. 30,31] giving the mean free path, with λ_s being the approximate sheath length.

$$\lambda = 0.12\left(\frac{1}{n_e(e^2/3kT)^2\ln(\lambda_s e^2/3kT)}\right) \tag{3}$$

Since the sheath extends at most a few mean-free lengths from the anode surface, curvature of the anode does not affect the governing equations in high pressure discharges. Thus, the region may be described in one dimension, the distance "y" from the anode surface. While the Debye length is sometimes assumed as the sheath extent, Reference 22 showed that the sheath thickness is a function of the anode fall voltage and the electron temperature. Equation (4) gives the appropriate form.

$$\lambda_s \approx \sqrt{\frac{\epsilon_o \Phi_a}{e n_\infty}} = \lambda_D \sqrt{\frac{e\Phi_a}{kT_e}} \tag{4}$$

An example case with a fall voltage of 100 volts gives a sheath extent of $\lambda_s = 2.352\times10^{-5}\ m$. This compares to a computed Debye length of $\lambda_D = 1.690\times10^{-6}\ m$. Therefore, the sheath can be an order of magnitude larger than the Debye length.

Nasser [Ref. 32] discusses an elementary theoretical approach to the glow discharge problem. He suggests a set of four one-dimensional ordinary differential equations, including the electron and ion current and number density equations, in addition to Poisson's equation. Most solution attempts have failed, with the boundary conditions being identified as the culprit. A similar attempt for the plasma thruster is discussed below.

2. Simplified Formulation

The steady probe equations are first written [Ref. 21] in their simplest form. The anode is assumed to operate as a heavily biased probe, which is true for low enough currents when the anode is not a source of ions. Whenever the temperature can be considered fixed, the energy equations are implicitly satisfied and, since ion inertia is neglected, the resulting set consists only of two species continuity equations and Poisson's equation. These equations are written in terms of y, which is the coordinate outward from the planar positive surface. Constants and variables are listed in Table 1.

TABLE 1 - NOMENCLATURE

a...characteristic length of plasma	n_∞...species number density at core plasma
$D_{i,e}$...species diffusion coefficient	N...total number density
e...elementary charge constant	T...temperature
E...electric field	T_o...neutral species temperature
E_o...electric field at anode surface	α...two-body recombination coefficient
E_∞...electric field at core plasma	ϵ_o...permittivity constant
$j_{i,e}$...species current density	ν...ionization coefficient
J...total current	$\mu_{i,e}$...species mobility coefficient
k...Boltzmann's constant	Φ_a...anode fall potential
K...current parameter	λ...mean-free distance
$n_{i,e}$...species number density	λ_d...Debye length
$\dot{n}_e$...time rate-of-change of n_e	λ_s...Sheath thickness

Note: Species subscripts denote ions (i) and electrons (e).

$$j_i = e\mu_i n_i E - (eD_i)\frac{dn_i}{dy} \tag{5}$$

$$j_e = e\mu_e n_e E + (eD_e)\frac{dn_e}{dy} \tag{6}$$

$$\frac{dE}{dy} = \frac{e}{\epsilon_o}(n_i - n_e) \tag{7}$$

$$J = j_i + j_e \tag{8}$$

$$\mu_{i,e} = \frac{eD_{i,e}}{kT_{i,e}} \tag{9}$$

Here, the j's are species contributions to the total current density. The existence of negative charges as free electrons is pivotal in the formulation. Next, the Einstein relation, equation (9), is introduced to write the mobilities in terms of the diffusion coefficients. We assume that the diffusion coefficients remain constant in the problem.

Equations (5) and (6) are next solved for $dn_{i,e}/dy$. The species current density equations are found from the net reaction rate of the plasma. Equations (10) and (11) combine to produce space derivatives for species current density.

$$\dot{n}_e = \nu_i n_e - \alpha n_i n_e \qquad (10)$$

$$\frac{dj_i}{dy} = \frac{-dj_e}{dy} = e\dot{n}_e \qquad (11)$$

Combining equations (5)-(11) produces a set of five coupled, non-linear differential equations describing the sheath. These are nondimensionalized to adjust all variables to the first order, and are rewritten below as equations (12)-(16), with nondimensionalized variables denoted by "$\hat{x}$". Nondimensionalization can be accomplished as follows: The species number densities n_e, n_i, are divided by their values at infinity to produce output from the anode surface to unity at the ambipolar boundary. The current densities j_e, j_i are divided by the total current, allowing the output to show the "mirror behavior" of the two currents. The electric field is divided by the initial anode value to give output starting from unity at the surface and decreasing to the final core field value. The variable "y" is divided by the characteristic length[5] of the plasma, "a", producing $\hat{y}$. These corrections allow all output to vary in the range from zero to one, as a function of distance from the anode.

[5]The characteristic length is defined so as to cancel the multiplying factor in the electric field equation, (14), ($a = 1.107 \times 10^{-6}$). This allows a physical interpretation of the ion/electron number densities, as well as the decay rate of the electric field.

$$\frac{d\bar{n}_i}{d\bar{y}} = \left(\frac{aeE_o}{kT_o}\right)\bar{n}_i\bar{E} - \left(\frac{aeE_\infty}{kT_o}\right)\bar{j}_i \tag{12}$$

$$\frac{d\bar{n}_e}{d\bar{y}} = -\left(\frac{aeE_o}{kT_o}\right)\bar{n}_e\bar{E} + \left(\frac{aeE_\infty}{kT_o}\right)\bar{j}_e \tag{13}$$

$$\frac{d\bar{E}}{d\bar{y}} = \left(\frac{aen_\infty}{E_o\epsilon_o}\right)(\bar{n}_i - \bar{n}_e) \tag{14}$$

$$\frac{d\bar{j}_e}{d\bar{y}} = -\left(\frac{akT_o v_i}{eE_\infty D_e}\right)\bar{n}_e\,(\bar{v}_i - \alpha\bar{n}_i) \tag{15}$$

$$\frac{d\bar{j}_i}{d\bar{y}} = \left(\frac{akT_o v_i}{eE_\infty D_e}\right)\bar{n}_e\,(\bar{v}_i - \alpha\bar{n}_i) \tag{16}$$

Attempts to solve this equation set using the computer code discussed below shows the set to be extremely sensitive to initial conditions. The computer code solver uses a "marching" scheme from the anode to the undisturbed plasma. The initial conditions are chosen to produce the electric field potential drop observed in actual thrusters. First and second space derivatives of the electric field are used as diagnostic checks to ensure reasonable output values and indicate instability of the integration process. Figure 4 shows the required resulting curves for the electric field and its first and second derivatives.

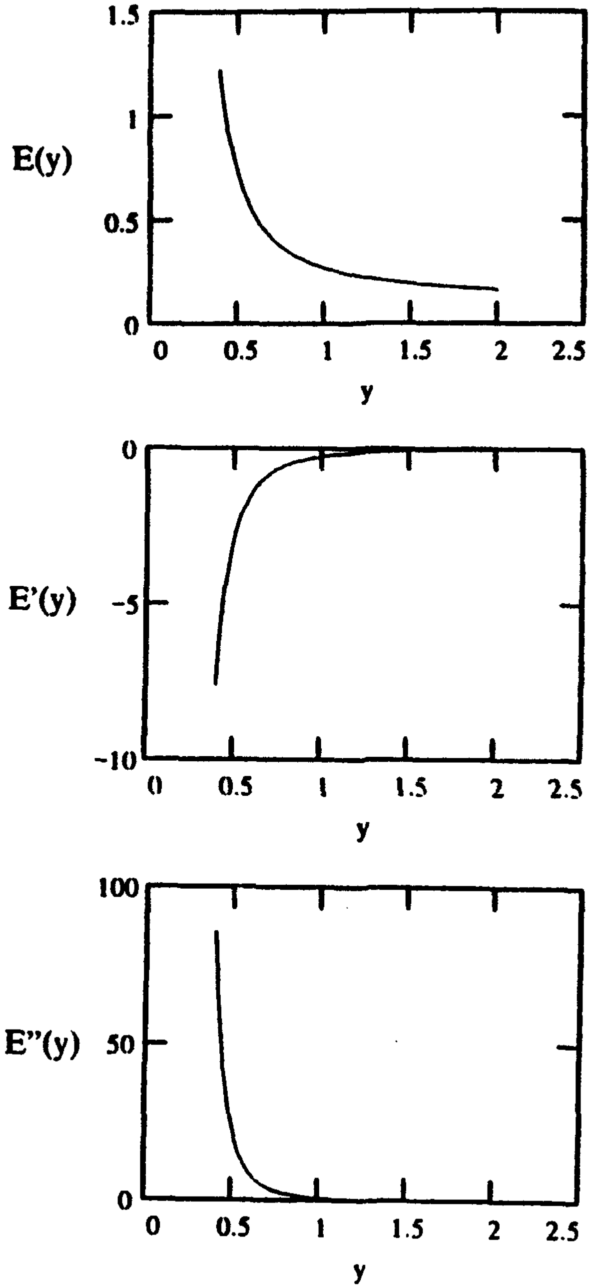

Figure 4 - Electric Field and First Two Space Derivatives Used as Diagnostic Checks for Integrator Output. (Plotted for a Generic Exponential Function).

Ecker characterizes the plasma at the anode as a double sheath, with the inner section called the "inertia sheath", and an outer section called the "energy loss section". The inner section shows a potential rise of the order of one volt, with the outer section showing the exponential potential drop shown in Figure 4. While this double sheath may in fact describe the actual sheath region, the formulation above only models the potential drop portion of the sheath, and does not attempt to produce the potential rise of the inner sheath. In addition, Ecker's current constrictions are of a "macroscopic" nature, whereas those of Reference 14 and this work are "microscopic" [Ref. 33].

Data for a 6,000°K Nitrogen plasma were used to test the equation set [Ref. 21]. Producing a proper solution required adjusting the initial conditions to force the curve shapes discussed above. Using Equations (2), (3), and (4), the mean free path, Debye length, and approximate sheath extent are calculated as $\lambda = 9.352\times10^{-3}$ m, $\lambda_D = 1.690\times10^{-6}$ m, and $\lambda_s = 2.352\times10^{-5}$ m (this assumes a drop voltage oı 100 volts).

3. Approximate Formulation

Reference 21 explores the above equation set by taking advantage of the symmetry among the equations, and introducing two parameters, K^+ and K^-, shown below.

$$K^+ \equiv \frac{j_i}{eD_i} + \frac{j_e}{eD_e} \tag{17}$$

$$-K^- \equiv \frac{j_i}{eD_i} - \frac{j_e}{eD_e} \tag{18}$$

It can be shown that the resulting equations can be manipulated to yield a single, ordinary differential equation for the K's in terms of the electric field. The resulting equation can be scrutinized for two distinct temperature regimes. Note that while the total current density, J, is constant in a steady, one-dimensional case, the K's can vary and will in turn also depend on the degree of reactivity of the plasma ($\dot{n}_e$), i.e.,

$$\frac{dj_i}{dy} = \frac{-dj_e}{dy} = e\dot{n}_e \tag{19}$$

Because ion diffusion is much slower than electron diffusion, it can be shown that the K's are related by

$$K^+ \approx -K^- + \frac{2J}{eD_e} \tag{20}$$

As will be evident, at the electrode surface the K's are equal to each other and at the undisturbed plasma, $K^- = 0$. The total current density may be evaluated from

$$J = en_{e}v_{e\infty} \qquad (21)$$

where $v_{e\infty}$ is the electron drift velocity beyond the ambipolar region which is strictly a function of E_{∞}/N, (i.e., of the ratio of undisturbed electric field to the total number density).

a. *Effects of Temperature on Anode Constriction*

It is useful to investigate the overall effects of temperature. Since temperature will be considered constant, it comes in as a parameter in this formulation whereas charge density and electric field remain as variables. Intuitive arguments will be introduced which suggest that the electron and ion/neutral temperatures play a rather singular role in determining the intrinsic dimensionality of the problem, (i.e., there are cases when the geometry of the current lines is not necessarily impressed by the electrode geometry). Since the problem is described by moderate pressure, largely collisional sheaths, the ion and neutral temperatures are anticipated to remain reasonably equal. Depending on the gas, the electron temperature, on the other hand, can be elevated from the gas temperature at the anode where actual magnitudes depend on the local value of E/N. In order to get a perspective on the effects of temperature, we shall consider two extremes, namely, the case where the electron and ion temperature are the same (the equilibrium case) and the case where the electron temperature is substantially elevated from that of the ions/neutrals (the two-temperature case).

(1) Case I: $T_e = T_i = T_o$ (Equilibrium)

The charge densities can be eliminated by combining equations (5)-(9), (17) and (18). The resulting equation can be shown to be

$$\frac{kT_o}{e}\left(\frac{K^*}{E}\right)' + K^* = \frac{2J}{eD_e} - \left(\frac{kT_o\epsilon_o}{e^2}\right)\frac{1}{E^2}\left[EE'' - (E')^2 - \frac{1}{4}\left(\frac{e}{kT_o}\right)^2 E^4\right]' \qquad (22)$$

If the electric field decreases monotonically from the wall to the undisturbed plasma (i.e., from $E_o \rightarrow E_\infty$), then as $y \rightarrow \infty$, $E \rightarrow E_\infty$, $E' \rightarrow 0$, $E'' \rightarrow 0$.

So that in equation (22) above the "outer solution" becomes:

$$K^* = \frac{2J}{eD_e} \qquad (23)$$

Now this represents an acceptable solution from a physical point of view. Moreover, as $y \rightarrow \infty$,

$$\dot{n}_{e\infty} \approx D_i(K^*)' \approx 0 \qquad (24)$$

which is also acceptable for an equilibrium situation at the undisturbed plasma. Results [Ref. 21] are shown in Figure 5 for the case of nitrogen at 6000°K using an approximate electric field distribution.

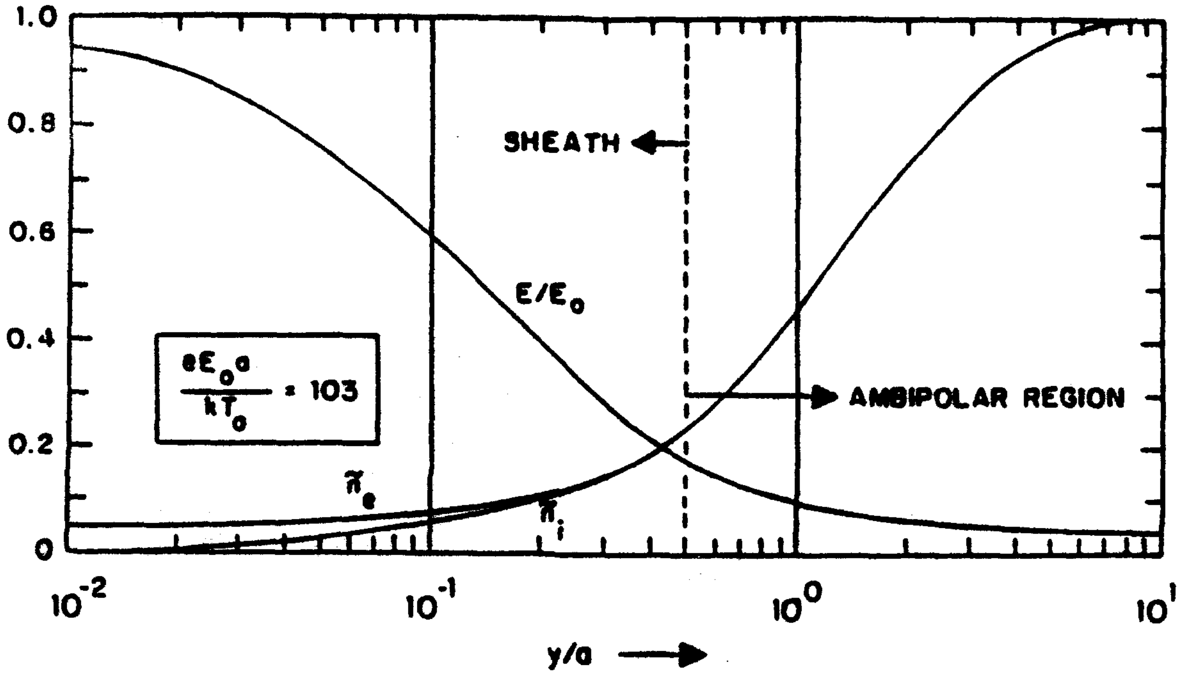

Figure 5 - Electric Field and Species as a Function of $\bar{y}$, Distance From Anode. An Approximation Using a Shaped Electric Field and Isothermal Plasma [Ref. 21].

(2) Case II: $T_e >> T_i = T_o$ (Two-Temperature)

In this case the same procedure as before yields the following equation where terms divided by T_e have been dropped when compared to their counterparts divided by T_o.

$$(K^*)' - \frac{K^*}{E} E' = \frac{2EJ}{kT_o D_e} + \frac{2\epsilon_o}{kT_o} EE'' + \frac{\epsilon_o}{eE} E''E' - \frac{\epsilon_o}{e} E''' \qquad (25)$$

Assuming the same monotonic decrease as before for the electric field from the wall to the plasma proper, as $y \to \infty$, $E \to E_\infty$, $E' \to 0$, $E'' \to 0$.

Then the outer solution becomes

$$\frac{dK^*}{dy} = \frac{2eEJ}{kT_o eD_e} \quad \text{with } \dot{n}_{e\infty} > 0 \qquad (26)$$

Or, $K^* \to$ (constant) y + constant, and $\dot{n}_{e\infty}$ keeps increasing with y.

This is <u>not</u> the proper outer solution for the one-dimensional, equilibrium plasma that we seek because the net ionization rate continues to increase well inside the plasma proper where conditions should saturate, yielding a constant electric field. Therefore, as formulated, Case II is not amenable to a one-dimensional solution. References 14 and 21 show how this case can be analyzed under a multidimensional approach. These references also discuss a method for describing the electron temperature as a function of E/N, then how to couple a simplified energy relation which satisfactorily describes a two-temperature plasma. The necessary ingredient to make equation (26) approach zero beyond the decrease of E to E_∞ is to allow J

to fan out as indicated in Figure 6. Thus, in equation (26), the product "EJ" can bring down the charge production rate to arbitrarily low values. Alternatively, it is possible to explore techniques of bringing the electron temperature down to be in closer equilibrium with the ions and neutrals. Transpiration cooling is one such means.

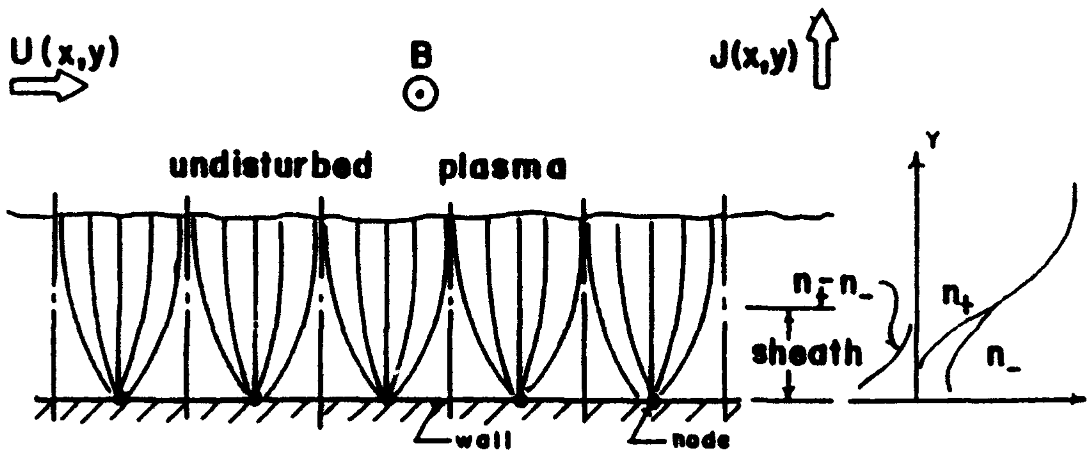

Figure 6 - Two-Dimensional Model of Current Paths Showing Periodic Structure. Thermal Instabilities and Inhomogeneities Would Favor One Site Over Others and a Single Macroscopic Constriction May Then Be Produced. [Refs. 14, 34]

b. *Similarity to Vacuum Arc Phenomena*

Instability phenomena observed in vacuum arcs [Ref. 35] are very similar to those observed in self-field thrusters [Ref. 12]. After the establishment of the current, the anode region operates in a vapor that issues from the electrodes. In vacuum arcs, Miller characterizes the anode region as operating in one of five distinct modes, ranging from a passive, low current mode to a high current, fully developed spot mode [Ref. 36]. Given the similarities mentioned above, vacuum arc anode research should be helpful in the understanding of MPD thruster transition to the anode spot mode. Existence diagrams after Miller [Ref. 36] are shown in Figure 7, which divide operating modes into regions as a function of anode current versus electrode geometry. Figure 7 shows the transition from glow to spot mode.

Anode spot formation at high currents is clearly a factor in limiting anode lifetime. Various phenomena have been related to anode spotting. Hugel [Ref. 12] relates the transition to spotting mode to an increase in $J^2/\dot{m}$ above a critical level. A separate factor connected with the spot mode is surface temperature of the anode. Rich, et.al., [Ref. 37] show that anode spotting is preceded by a luminous "footpoint" and followed by local melting prior to spot formation. Separately, Schuocker [Ref. 38] finds a connection between spotting initiation and the factors of anode evaporation and magnetic constriction in vacuum arcs with high currents. Experimental investigations must be performed to see if the above-mentioned vacuum arc criteria apply to self-field thrusters.

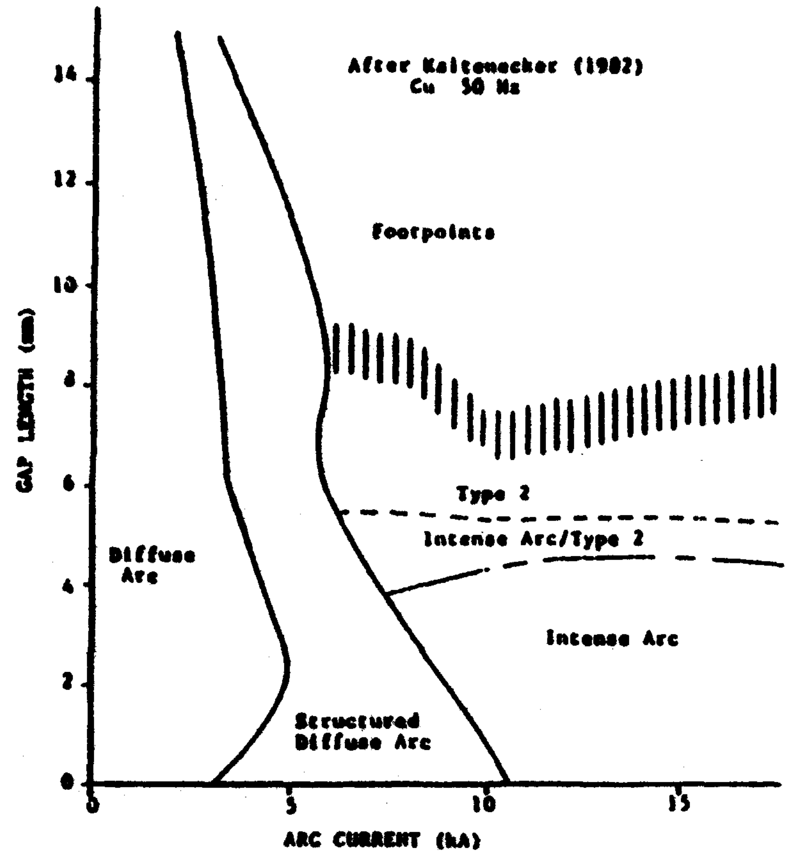

Figure 7 - Anode Discharge Modes as a Function of Current and Gap Length. [Ref. 36]

C. COMPUTER CODE

Rather than using linear approximations to equations (12)-(16), the nonlinear set was used, with initial conditions adjusted in an attempt to produce observed electric fields from probe data. First and second space derivatives of the electric field were used as diagnostic checks to ensure computed output was reasonable. Initial conditions computed from the approximate formulae in Reference 21 were used. The equation set above presents a difficult problem for two reasons, nonlinearity and multiple time constants. The species number density equations, (12) and (13), both contain a nonlinear term, each with a time constant of its own. In addition, the electric field equation, (14), adds a possible third time constant. This constitutes a "stiff" set of equations. Attempts were made to solve the set with the data discussed above, using Gear's method of backward differentiation, in hopes that the variables would change slowly enough with each iteration to render a convergent iterative process. As described in Reference 39, if some reactions are slow and others fast among a set of coupled equations, the fast ones will control the stability of the method. This is addressed in the DGEAR program available from the International Mathematical & Statistical Library (IMSL). The latter software contains an Adams predictor-corrector method, as well as Gear's method, which is well known for its success at solving stiff equation sets. The DGEAR software allows for a choice of functional or chord iteration methods, as well as a choice of Jacobian matrices. A more detailed discussion of this software can be found in Reference 39 and in the IMSL library. [Ref. 39]

D. COMPUTATIONAL RESULTS

Numerous computer runs were completed using the initial conditions taken from Reference 21. In addition, data for the ionization coefficient ν, Figure 8, was taken from References 40 and 41.

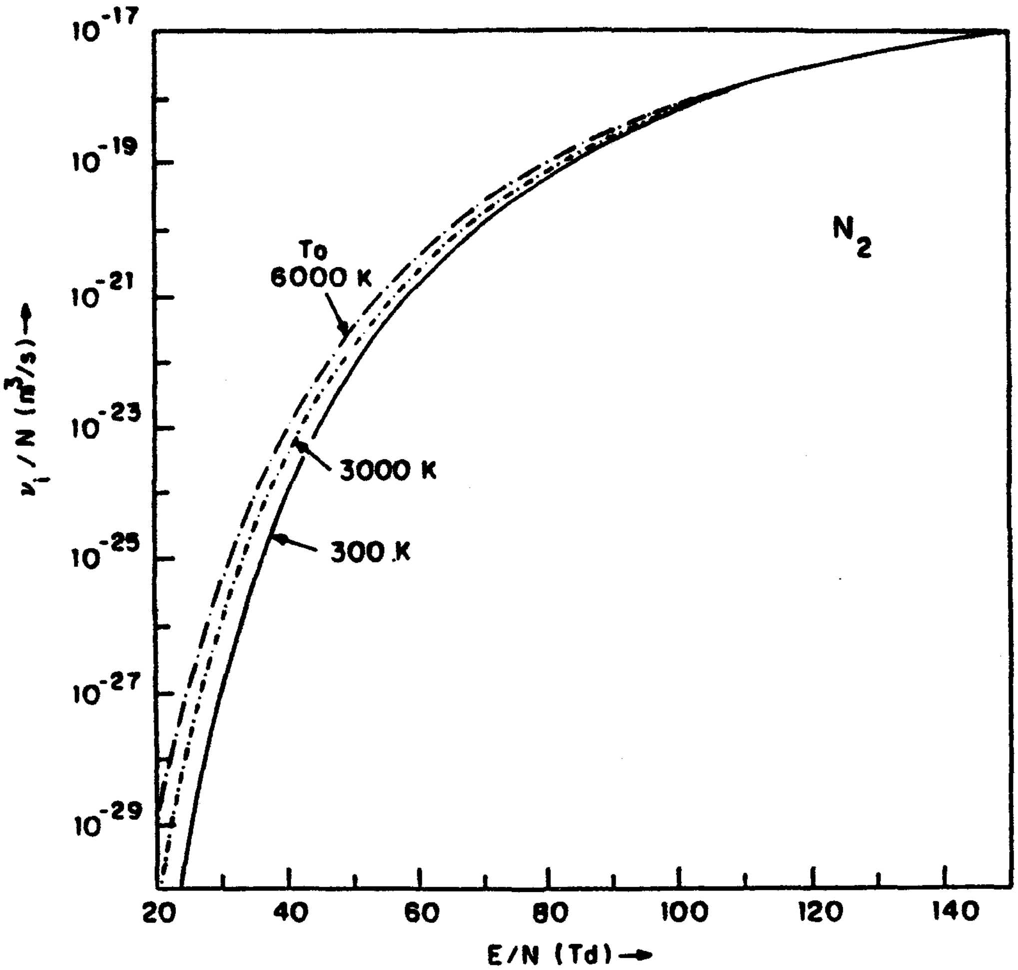

Figure 8. Ionization Coefficient ν as a Function of E/N for Nitrogen for Various Vibrational Temperatures (Refs. 40,41).

Various combinations of initial conditions and ionization coefficient were used. As mentioned above, the electric field and its space derivatives were used as

diagnostic/reasonability checks on the output. Individual, as well as multiple computer runs were attempted to model the sheath region. Nonlinearities in the equation set are clearly seen in Figure 9. The ion number density does not reach that of the electrons, and the latter population growth rate continues to grow without bound. The shape of the electron population curve is very sensitive to its initial value. As shown in Figure 9, the latter population has too high a growth rate when compared to the ion population, and the latter does not "catch up". Increasing the initial value of $\tilde{n}_e$ flattens out this curve to a reasonable shape. Above an initial value of approximately 0.06, however, the plot of $\tilde{n}_e$ "dips" after a certain distance and then continues to increase as expected. This gives an approximate upper value for this initial value. To avoid instabilities like this, small "slices" were taken of the output after a small number of integration steps and multiple runs were used to form a "cut and paste" plot of the region. When a reasonable plot shape was produced, the value of ionization coefficient was varied in the "slices" to attempt to produce the required end values for electric field and species population. Both multiple and equilibrium values for the ionization coefficient were used. When the data showed signs of instability and failure to follow the required forms of Figure 4, a "slice" was made in the data stream, and the data points from this point used to start a new computer run. This approach was taken in the hope of avoiding singularities in the integration from anode surface to ambipolar region. In addition to the diagnostic checks shown in Figure 4, an additional data check is provided by the transition from the sheath to the ambipolar region.

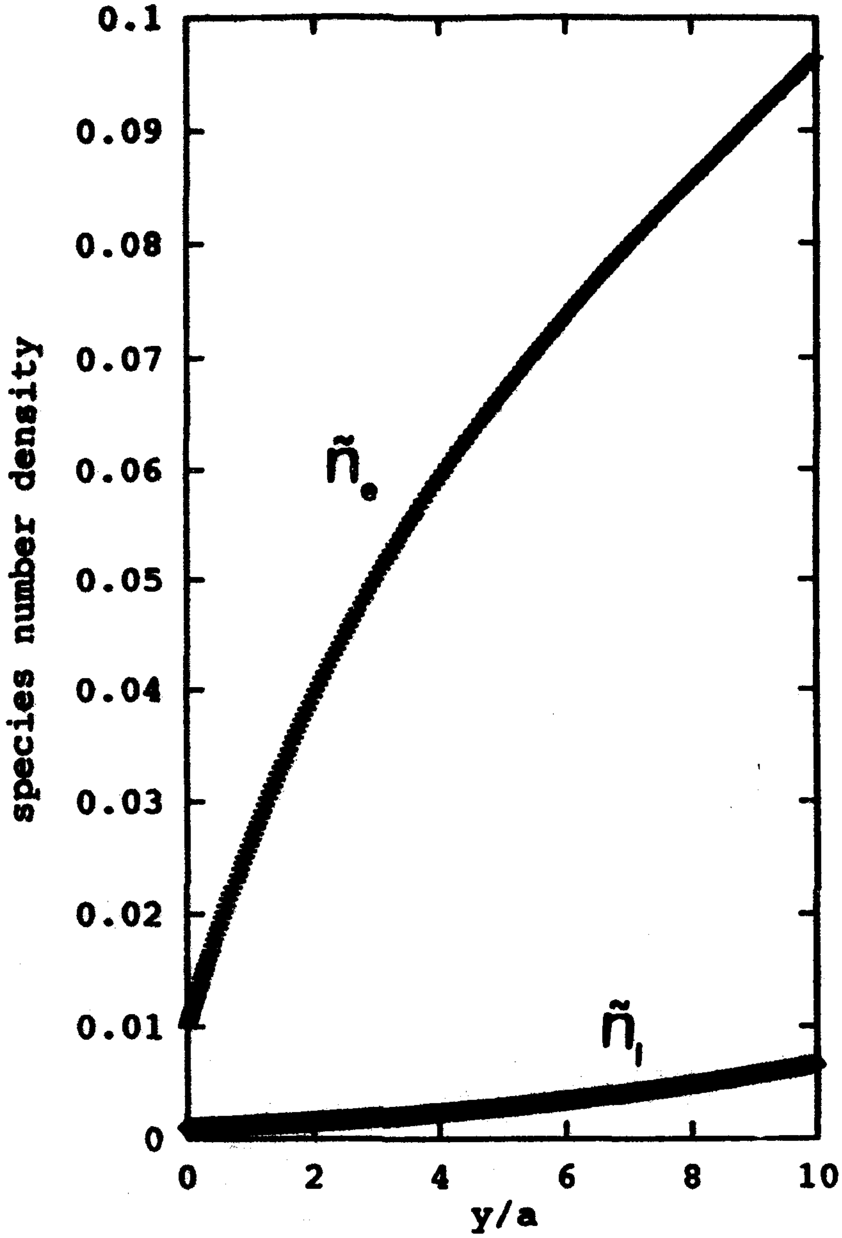

Figure 9. Species Number Density plots for Individual Computer Run, Showing Divergent Tracks for Ion and Electron Populations, and Effect of Nonlinearity.

As shown in Figure 5, the species number densities are equivalent in this region, as are their change rate. Thus, setting Equations (12) and (15) equal to each other and solving for $\tilde{n}$ yields a value of 0.5 in the ambipolar region. As indicated in Figures

10-11, the output produces the desired plot slopes for electric field and species number density. However, the number density plots cross long before approaching the required value of 0.5. In addition, neither electric field nor species number density approaches an equilibrium value or shows sign of levelling off. Apparently, the multiple time constants and nonlinear portions of the number density equations combine to create a seemingly intractable system. Solutions for this system may be possible for specific, individual initial condition sets, but the problem does not appear amenable to this approach in general. A one-dimensional system such as this may be better described through the approach of boundary layer theory or nonlinear dynamics and chaos. Given the effort and difficulty involved in the latter, a one-dimensional approach such as that modelled above does not appear useful. A combination of one- and two-dimensional modelling would appear to be more useful, as discussed in Reference 14. A one-dimensional model may be useful, but only in an approximation approach, with a shaped electric field, such as that used in Reference 21.

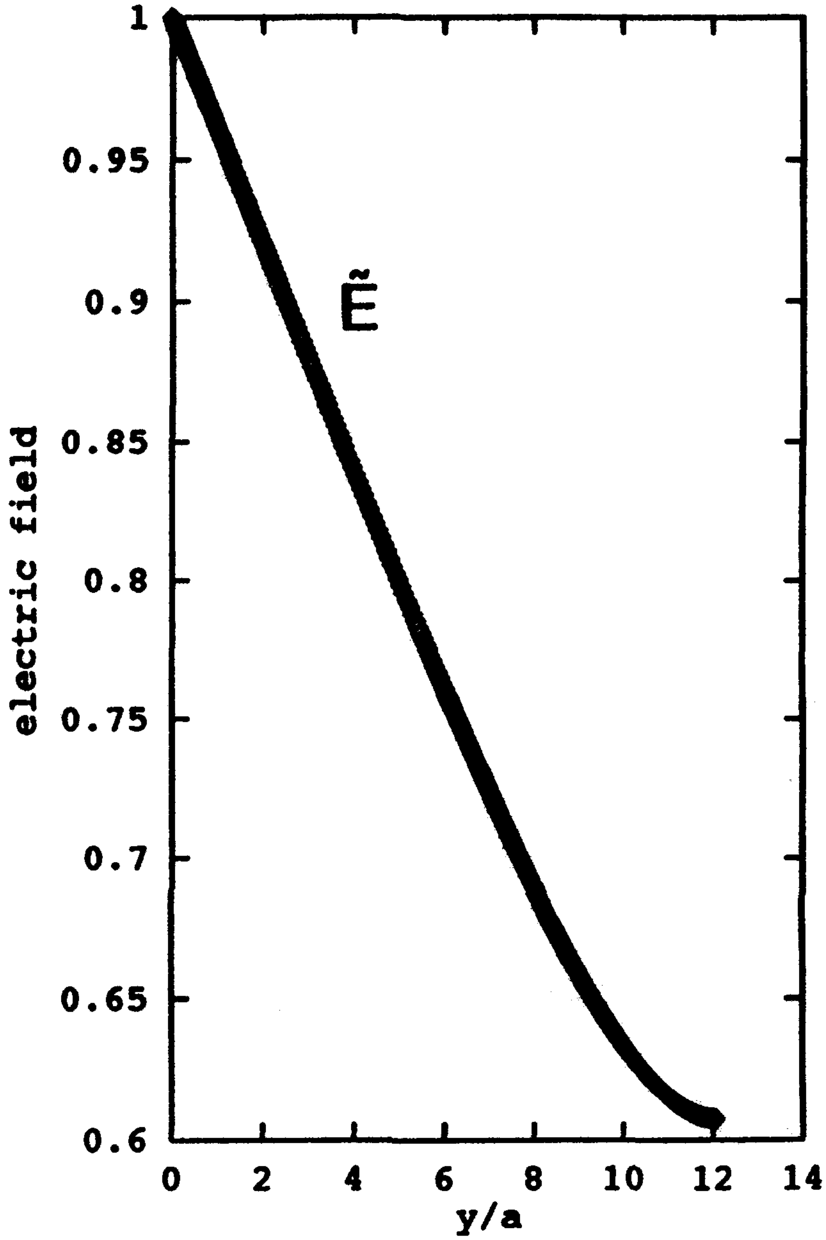

Figure 10. Electric Field as a Function of $\tilde{y}$, Distance From Anode, Using Equations (12)-(16).

ionization (with some help from the tail of the Maxwellian distribution of electron energies), what results is a breakdown voltage appreciably below the ionization potential. This then could be an explanation for the low voltage breakdown observations [Ref. 42]. Clearly, gases with low ionization potentials and lots of atomic electron energy levels are preferred (such as cesium and barium) but low-voltage breakdown has been observed with most gases.

The increase of the anode fall voltage above the ionization potential has been related to the electron Hall parameter, since a reduction of this parameter decreases that voltage and corresponding losses. Control of the local magnetic field through the use of and array of permanent magnets as well as implementation of transpiration cooling (which increases the electron collision frequency) have both yielded some encouraging results. Because the anode fall also scales up with $J^2/\dot{m}$, it is conceivable that current inhomogeneities and plasma instabilities which are reflected in this parameter are in the picture as well. [Ref. 44]

In summary, any possible reduction of the anode fall voltage will hinge on a thorough understanding of the anode region, with its associated sheath and ambipolar regions, where electron temperature effects, ionization effects, and magnetic field effects play a pivotal role. If transpiration cooling is present, then additional phenomena of fluid-dynamic nature may come into play. Experimental observations with atmospheric discharges indicate the possible presence of convective effects at the anode. [Ref. 45]

IV. TRANSPIRATION COOLING

Transpiration cooling of the anode has often been promoted as an attractive means of recovering a large portion of the power deposited there. Additionally, the onset of melting may be minimized or even avoided by active anode cooling. Rich, et.al., related high anode temperatures to anode spotting [Ref. 37]. Similarly, Park and Choi showed that low thermal diffusion leads to erosion and, consequently, anode damage [Ref. 46]. Active anode cooling via transpiration is one means of ensuring high thermal diffusion and extending anode lifetime. Early work by Schoeck, et. al., [Ref. 47] showed that up to 80% of the energy deposited in the anode is recoverable via transpiration cooling. While this study used non-convective, high-intensity argon arcs, it is reasonable to assume that this effect would apply in MPD arcs using other propellants. Although this cooling method has not been studied for incorporation in plasma thrusters for some time, it has been recently considered as a means of cooling the fuselage of the National Aerospace Plane (NASP). Plasma thruster designs could undoubtedly gain from this database, and due consideration should be given to this cooling approach for the anode.

For a given mass transfer flow rate, the heat flux reduction to a surface is inversely proportional to the molecular weight of the injected gas. Use of the propellant as coolant as well as fuel would eliminate the need for additional tankage and pumps, simplifying the design considerably. Lithium has been considered to be

the propellant of choice, primarily because of its low molecular weight, its favorable ionization potential, and its low-volume tankage properties. It has a relatively low first ionization potential of 5.4 eV and a high second ionization potential of 75.6 eV. This single ionization potential range of over 70 eV compares to approximately 20 eV for Cesium and 27 eV for Potassium [Ref. 48]. This provides a broad temperature range within which only single ionization will occur. Large temperatures must be reached within the gas before double ionization occurs. As the gas temperature is increased several thousand degrees Kelvin, it undergoes ionization and disassociation. Thermal energy deposited can be recovered through nozzle expansion at the exit. However, residence time of the gas is not long enough to ensure recombination. Thus, the energy invested in ionization and disassociation will be lost. [Ref. 49] Lithium has been shown to produce specific impulse figures in excess of 7,000 seconds at 70% efficiency in steady-state thrusters, [Ref. 50] whereas all other propellants have been limited to less than 3,000 seconds specific impulse at less than 40% efficiency [Ref. 6]. Subramaniam has concluded that:

> ...regenerative cooling of anodes (at the specific impulse values in the MPD regime) is possible only with hydrogen or with alkali-metal propellants, notably lithium. In the latter case, the ideal anode operating mode would be evaporation and ionization of the propellant on the porous or wetted anode surface, resulting in increased ion current fraction, reduced anode voltage fall and utilization of part of the anode loss energy [Ref. 51].

Liquid coolants, as well as reducing storage requirements, offer the advantage of providing latent heat of vaporization for energy disposal. However, design problems can occur if the liquid is allowed to vaporize within the porous structure. Problems

arise due to the abrupt increase in pressure gradient as the coolant vaporizes. Since coolant flow generally have three-dimensional characteristics, the flow will be diverted around the vapor bubbles and hot spots often develop. The technical practicality of using molten lithium to cool a porous tungsten anode would seem to be beyond current technology. On the other hand, the products of decomposition of hydrazine (gaseous hydrogen and ammonia) have proven to be efficient and practical coolants [Ref. 52].

Given the performance figures above, using an auxiliary coolant gas even with high molecular weight (e.g., NH_3, N_2, CH_4, etc.) which could serve as a propellant once released from a porous hot tungsten anode surface would seem to more practical, vice dealing with molten lithium. Experimental studies would be needed to compare the approaches. Kuriki and Suzuki performed experiments with a quasi-steady MPD thruster to study the effect of anode gas injection (Argon). At high currents of up to 10 kA, increases in thrust, specific impulse, and flow discharge stability were observed [Ref. 53].

There is some question as to the likelihood of current constriction resulting from anode gas injection. In such a case, swirling or circulating the propellant gas would help to move any footpoints that developed around the anode surface and prevent them from becoming fully-developed spots. Additionally, an applied magnetic field could serve to circulate the footpoints as well. The unique advantage of transpiration cooling hinges on providing effective anode cooling while supplying

hot propellant, but the real benefit will depend on how small the amount of coolant required really will be.

Transpiration cooling has proven to be as desirable as it is challenging. It is complicated to implement, with associated reliability problems and difficulty of analytical predictions. While the production of thicker boundary layers is largely ineffective against the electron flux heating, the cooling itself is most efficient and a substantial fraction of the energy transferred to the anode is recoverable. The arguments of Chapter Three indicate that a reduction of the electron temperature in the anode would have the desirable effect of reducing the initial current spotting which can be conjectured to be the path that leads to anode arc spots. This electron temperature reduction can be done most effectively by polyatomic gases (which have a high δ-loss factor) emanating from the anode surface [Ref. 54].

The arguments relating to transpiration cooling might be summarized as follows:

<u>Favorable Outcomes</u>

- No separate cooling mechanism for anode required,
- Adds "hot" propellant to exhaust "recovering" most of the electrical power loss to the anode,
- Quenches T_e thus likely to postpone anode spotting <u>and</u> reduce the heating associated with the electron thermal energy ($5kT_e/2e$),
- Reduces bulk convective heating,
- Reduces the local electron Hall parameter by increasing the collision frequency,

Favorable Outcomes (cont'd.)

- Allows for some radiation cooling from the hot tungsten surface (about 120 watts/cm^2 at 2800°K [Ref. 51]),
- Hydrogen/ammonia gases flowing through hot porous (sintered) tungsten represent a compatible, proven technology.

Unfavorable Outcomes

- May decrease the electrical conductivity in the anode region,
- May destabilize the ionization processes in the sheath and bring about significant fluctuations in the current,
- Disrupts "cathode jet" in front of the anode with unpredictable consequences,
- Introduces propellant which may not be hot enough, not ionized enough, or not in the proper place for $\vec{J} \times \vec{B}$ acceleration,
- Transpiration cooling through a porous (tungsten) anode is a difficult design problem.

V. CONCLUSIONS AND RECOMMENDATIONS

Plasma thrusters offer distinct advantages in terms of payload delivered for interplanetary missions, as well as for orbital transfer. A recent comparison completed by Choueiri, Kelly, and Jahn shows a mass savings of 65 tons for an orbital transfer from low Earth orbit to geosynchronous Earth orbit using a quasisteady MPD thruster as opposed to an advanced chemical thruster.[6] This superior performance comes at the expense of low thrust-to-weight ratio and long transit time. However, given the large cargo/logistic requirements of a manned interplanetary mission, delivery of payload must be maximized. Thus, further work to characterize more fully thruster behavior and anode contributions in particular are certainly warranted. [Ref. 55]

The "cut-and-paste" method used to generate Figures 10 and 11 is not of practical use as a modelling method, due to the large effort involved. It did produce the expected electric field and species number and current density plots near the anode, but failed to produce the entire sheath out to the ambipolar region. The nonlinearity of the equation set led to a quickly deteriorating solution. A more practical approach using nonlinear dynamics and/or chaos must be developed to model the sheath numerically.

---

[6]This assumes a specific impulse of 2,000 seconds, 600 kW of input power, and a 270-day transit time.

Depending on the propellant mass fraction used for cooling, the transpiration scheme discussed above presents some rather unique advantages. A hot anode which uses only a small amount of propellant for cooling need not be penalized for any lost thrust. If in addition, we increase anode lifetime by delaying the formation of anode arc spots, then the scheme is all the more desirable. A decrease of the electron temperature in the vicinity of the anode may bring about a more homogeneous flow of current and a reduction in the heating effect associated with the high electron kinetic energy. Recovery of the heat deposited at the anode would be most important if the propellant fraction is high. In such case, nozzle expansion of the hot-propellant/coolant-gas might be implemented.

Means of limiting anode losses through decreasing anode fall voltages were discussed, including the control of the local Hall parameter and the implementation of thermionic arc breakdown. The electrical conductivity (of a nonreacting plasma) could possibly decrease as a result of transpiration cooling and this might increase the anode fall voltage.

Additional work needs to be done in the following areas:

- Investigate effectiveness of nonlinear dynamics and chaos in solving sheath equation set,
- Incorporate adequate one- or two-dimensional sheath modelling in quasisteady MPD numerical codes,

- Investigate the role that fluid dynamic effects play in MPD thruster anode discharges,
- Investigate the effect of transpiration cooling on current and plasma stability, as well as on thruster performance and lifetime,
- Determine effectiveness of transpiration cooling's increase of the collision frequency parameter,
- Compare performance of gaseous propellant/coolants versus hybrid designs with lithium propellant/gaseous coolant,
- Determine if required percentage of propellant gas as coolant is practical (e.g., less than 10%),
- Investigate effect of surface imperfections as focal points for current constrictions and as precursors to anode spotting.

APPENDIX A

The following software includes the calling program, SHEATH, its two subroutines, FCNJ and YDOT, and the DGEAR integrator. The latter is quite extensive in length and includes ten subroutines, including the following: DGRST, DGRCS, DGRPS, DGRIN, LUDATF, LUELMF, LEQT1B, UERTST, UGETIO, and USPKD. A detailed discussion may be found in IMSL literature or Reference 39.

```
*******************************************************************************
      Program Sheath                                                    000010
C                                                                       000020
C-----Calling program for DGEAR integrator. Initial conditions are      000030
C     input via READ statements and keyboard entry. Output is to data   000040
C     files via the DGRST subroutine. Diagnostic check of output via    000050
C     Figure 4 printed to data file from DGRST subroutine. Consult      000060
C     DGEAR comments for variable descriptions not listed below.        000061
C--------------------                                                   000062
      REAL E,K,EPS,TI,EF,EFINF,DI,DE,NUINF,C1,K1,A                      000070
      INTEGER N,METH,MITER,INDEX,IWK(1),IER,STEP                        000080
      REAL*8 X,H,Y(5),XEND,TOL,WK                                       000090
      EXTERNAL YDOT, FCNJ, DGEAR                                        000100
      COMMON/CONST/E,K,EPS,TI,EF,EFINF,NINF,DI,DE,VINF,C1,K1,A          000110
C                                                                       000120
C------------Constants                                                  000121
C     C1 and K1 are constants describing the ionization coefficient.    000122
C     They are taken from the data plotted in Figure 8. The             000123
C     coefficient is equal to the nondimensionalized electric potential 000124
C     raised to the K1 power and then multiplied by C1.                 000125
C     In this way, the ionization coefficient is allowed to vary in     000126
C     proportion to the strength of the electric potential.             000127
C-------------                                                          000128
      WRITE(*,*)'Input value for C1 (format 6E3):'                      000129
      READ(*,*)C1                                                       000130
      WRITE(*,*)C1                                                      000131
      WRITE(*,*)'Input value for K1 (format 6E3):'                      000132
      READ(*,*)K1                                                       000133
      WRITE(*,*)K1                                                      000134
C     Initial conditions for species number density, electric potential 000135
C     and species current density are now input (ni,ne,E,je,ji).        000136
C--------------                                                         000137
      WRITE(*,*)'Input values for y(1) through y(5) (format 5(6E3)):'   000138
      READ(*,*)y(1),y(2),y(3),y(4),y(5)                                 000139
      WRITE(*,*)y(1),y(2),y(3),y(4),y(5)                                000140
C     Following constants are for plasma described in Reference 21      000141
C     (6,000 K, Init E=20,000 V/m, Final E=1,200 V/m)                   000142
      E=1.6E-19                                                         000143
      K=1.38E-23                                                        000150
      EPS=8.854E-12                                                     000160
      TI=6E3                                                            000170
      EF=2E5                                                            000180
      EFINF=1.2E4                                                       000190
      DI=1.724E-4                                                       000200
      DE=1.724E-1                                                       000210
      VIO = 2.E6                                                        000215
      VINF = 4.93E-7                                                    000220
C                                                                       000230
```

```
C     A is plasma characteristic length which shows potential drop.
C     A = ((EPS*EF)/(E*NINF)) = 1.107E-6
      A = 1.107E-5
      X = 0.01
      XEND = 10.
      H = 1e-6
      TOL = 1E-6
      METH = 2
      MITER = 1
      INDEX = 1
      N=5
      IWK(1) = 5
      WK = 18000.
      IER = 0
      OPEN(UNIT=8,FILE='SHEATH.DAT',STATUS='UNKNOWN')
      CALL DGEAR2(N,YDOT,FCNJ,X,H,Y,XEND,TOL,METH,MITER,INDEX,IWK,WK,
     +IER,STEP)
      DO 3 I=0,N
   DO 2 J=0,100
      WRITE(*,*)J,Y(I)
      WRITE(8,1)J,Y(I)
    1 FORMAT(T2,F5.1,5(5X,D9.2))
    2   CONTINUE
    3 CONTINUE
      WRITE(*,*)'Total Steps = ',STEP,'Final Step Size = ',H,
     +'Error Code = ',IER
      CLOSE(UNIT=8)
      STOP
      END
C************************************
C DUMMY SUBROUTINE FCNJ
C************************************
      SUBROUTINE FCNJ(N,X,Y,PD)
      INTEGER N
      REAL Y(N),PD(N,N),X
      RETURN
      END
C************************************
C SUBROUTINE YDOT
C************************************
      SUBROUTINE YDOT(N,X,Y,YPRIME,eprime,eprime2)
      REAL*8 X, Y(5),YPRIME(5),NUI,eprime,eprime2
       REAL E,K,EPS,TI,EF,EFINF,NINF,DI,DE,VINF,C1,K1,A,B1,B2,B3,B4
        COMMON/CONST/E,K,EPS,TI,EF,EFINF,NINF,DI,DE,VIO,VINF,C1,K1,A
      VI = C1 * (Y(3)**K1)
      VIT = VI / VIO
C     Following constants are the bracketed values in Equations 12-16.
C     A is left as a variable.
C-----------------
C     B1 = ((E*EPS)/(K*TI)) * A
      B1= 3.86E5 * A
C     B2 = ((E*EFINF)/(K*TI)) * A
      B2 = 2.32E4 * A
C     B3 = ((E*NINF)/(EF*EPS)) * A
      B3 = 9.04E5 * A
C     B4 = ((VINF*K*TI)/(E*DE*EFINF)) * A
      B4 = 2.62E-21 * A
C     Alpha = 2-body recombination coefficient (fm. Laser Kinetics
C     Handbook (AFWL-TR-74-216, 1974)) (cm3/sec)
      Alpha = 9.e-8
```

```
C-------------------------
 C FIVE FIRST ORDER EQUATIONS - Equations 12-16
C-------------------------
C      Ni
       YPRIME(1) =   (B * Y(1) * Y(3)) - Y(5)
C      Ne
       YPRIME(2) = -(B * Y(2) * Y(3)) + Y(4)
C      E
       YPRIME(3) =  B3 * (Y(1) - Y(2))
C      je
       YPRIME(4) = -B4 * Y(2) * (VIT - (ALPHA * Y(1)))
C      ji
       YPRIME(5) =  B4 * Y(2) * (VIT - (ALPHA * Y(1)))
C
C---Diagnostic Check of first,second derivatives-----
C
       eprime = y(1) - y(2)
       eprime2 = yprime(1) - yprime(2)
C
       RETURN
       END
```

```
C   IMSL ROUTINE NAME   - DGEAR
C
C--modified to return # of steps via variable "step" in subroutine call +
C
C   COMPUTER            - IBM/DOUBLE
C
C   LATEST REVISION     - NOVEMBER 1, 1984
C
C   PURPOSE             - DIFFERENTIAL EQUATION SOLVER - VARIABLE ORDER
C                           ADAMS PREDICTOR CORRECTOR METHOD OR
C                           GEARS METHOD
C
C   USAGE               - CALL DGEAR (N,FCN,FCNJ,X,H,Y,XEND,TOL,METH,
C                           MITER,INDEX,IWK,WK,IER)
C
C   ARGUMENTS    N      - INPUT NUMBER OF FIRST-ORDER DIFFERENTIAL
C                           EQUATIONS.
C                FCN    - NAME OF SUBROUTINE FOR EVALUATING FUNCTIONS.
C                           (INPUT)
C                         THE SUBROUTINE ITSELF MUST ALSO BE PROVIDED
C                           BY THE USER AND IT SHOULD BE OF THE
C                           FOLLOWING FORM
C                             SUBROUTINE FCN (N,X,Y,YPRIME)
C                             REAL X,Y(N),YPRIME(N)
C                                    .
C                                    .
C                                    .
C                         FCN SHOULD EVALUATE YPRIME(1),...,YPRIME(N)
C                           GIVEN N,X, AND Y(1),...,Y(N). YPRIME(I)
C                           IS THE FIRST DERIVATIVE OF Y(I) WITH
C                           RESPECT TO X.
C                         FCN MUST APPEAR IN AN EXTERNAL STATEMENT IN
C                           THE CALLING PROGRAM AND N,X,Y(1),...,Y(N)
C                           MUST NOT BE ALTERED BY FCN.
C                FCNJ   - NAME OF THE SUBROUTINE FOR COMPUTING THE
C                           JACOBIAN MATRIX OF PARTIAL DERIVATIVES.
C                           (INPUT)
C                         THE SUBROUTINE ITSELF MUST ALSO BE PROVIDED
C                           BY THE USER.
C                         IF MITER=1 IT SHOULD BE OF THE FOLLOWING
C                           FORM
C                             SUBROUTINE FCNJ (N,X,Y,PD)
C                             REAL X,Y(N),PD(N,N)
C                                    .
C                                    .
C                         FCNJ MUST EVALUATE PD(I,J), THE PARTIAL
C                           DERIVATIVE OF YPRIME(I) WITH RESPECT TO
C                           Y(J), FOR I=1,N AND J=1,N.
C                         IF MITER= -1 IT SHOULD BE OF THE FOLLOWING
C                           FORM
C                             SUBROUTINE FCNJ (N,X,Y,PD)
C                             REAL X,Y(N),PD(1)
C                                    .
C                                    .
C                         FCNJ MUST EVALUATE PD IN BAND STORAGE MODE.
C                           THAT IS, PD(N*(J-I+NLC)+I) IS THE PARTIAL
C                           DERIVATIVE OF YPRIME(I) WITH RESPECT TO
C                           Y(J). NLC IS THE NUMBER OF LOWER
C                           CODIAGONALS FOR THE BAND MATRIX.
C                         FCNJ MUST APPEAR IN AN EXTERNAL STATEMENT IN
C                           THE CALLING PROGRAM AND N,X,Y(1),...,Y(N)
```

```
C                      MUST NOT BE ALTERED BY FCNJ.                 DGEA0620
C                    FCNJ IS USED ONLY IF MITER IS EQUAL TO         DGEA0630
C                      1 OR -1. OTHERWISE A DUMMY ROUTINE CAN       DGEA0640
C                      BE SUBSTITUTED. SEE REMARK 1.                DGEA0650
C          X       - INDEPENDENT VARIABLE. (INPUT AND OUTPUT)       DGEA0660
C                    ON INPUT, X SUPPLIES THE INITIAL VALUE         DGEA0670
C                      AND IS USED ONLY ON THE FIRST CALL.          DGEA0680
C                    ON OUTPUT, X IS REPLACED WITH THE CURRENT      DGEA0690
C                      VALUE OF THE INDEPENDENT VARIABLE AT WHICHDGEA0700
C                      INTEGRATION HAS BEEN COMPLETED.              DGEA0710
C          H       - INPUT/OUTPUT.                                  DGEA0720
C                    ON INPUT, H CONTAINS THE NEXT STEP SIZE IN     DGEA0730
C                      X. H IS USED ONLY ON THE FIRST CALL.         DGEA0740
C                    ON OUTPUT, H CONTAINS THE STEP SIZE USED       DGEA0750
C                      LAST, WHETHER SUCCESSFULLY OR NOT.           DGEA0760
C          Y       - DEPENDENT VARIABLES, VECTOR OF LENGTH N.       DGEA0770
C                     (INPUT AND OUTPUT)                            DGEA0780
C                    ON INPUT, Y(1),...,Y(N) SUPPLY INITIAL         DGEA0790
C                      VALUES.                                      DGEA0800
C                    ON OUTPUT, Y(1),...,Y(N) ARE REPLACED WITH     DGEA0810
C                      A COMPUTED VALUE AT XEND.                    DGEA0820
C          XEND    - INPUT VALUE OF X AT WHICH SOLUTION IS DESIRED DGEA0830
C                    NEXT. INTEGRATION WILL NORMALLY GO             DGEA0840
C                    BEYOND XEND AND THE ROUTINE WILL INTERPOLATEDGEA0850
C                      TO X = XEND.                                 DGEA0860
C                  NOTE THAT (X-XEND)*H MUST BE LESS THAN           DGEA0870
C                    ZERO (X AND H AS SPECIFIED ON INPUT).          DGEA0880
C          TOL     - INPUT RELATIVE ERROR BOUND. TOL MUST BE        DGEA0890
C                    GREATER THAN ZERO. TOL IS USED ONLY ON THE     DGEA0900
C                    FIRST CALL UNLESS INDEX IS EQUAL TO -1.        DGEA0910
C                    TOL SHOULD BE AT LEAST AN ORDER OF             DGEA0920
C                    MAGNITUDE LARGER THAN THE UNIT ROUNDOFF        DGEA0930
C                    BUT GENERALLY NOT LARGER THAN .001.            DGEA0940
C                    SINGLE STEP ERROR ESTIMATES DIVIDED BY         DGEA0950
C                    YMAX(I) WILL BE KEPT LESS THAN TOL IN          DGEA0960
C                    ROOT-MEAN-SQUARE NORM (EUCLIDEAN NORM          DGEA0970
C                    DIVIDED BY SQRT(N)). THE VECTOR YMAX OF        DGEA0980
C                    WEIGHTS IS COMPUTED INTERNALLY AND STORED      DGEA0990
C                    IN WORK VECTOR WK. INITIALLY YMAX(I) IS        DGEA1000
C                    THE ABSOLUTE VALUE OF Y(I), WITH A DEFAULT     DGEA1010
C                    VALUE OF ONE IF Y(I) IS EQUAL TO ZERO.         DGEA1020
C                    THEREAFTER, YMAX(I) IS THE LARGEST VALUE       DGEA1030
C                    OF THE ABSOLUTE VALUE OF Y(I) SEEN SO FAR,     DGEA1040
C                    OR THE INITIAL VALUE OF YMAX(I) IF THAT IS     DGEA1050
C                     LARGER.                                       DGEA1060
C          METH    - INPUT BASIC METHOD INDICATOR.                  DGEA1070
C                    USED ONLY ON THE FIRST CALL UNLESS INDEX IS DGEA1080
C                     EQUAL TO -1.                                  DGEA1090
C                  METH = 1, IMPLIES THAT THE ADAMS METHOD IS       DGEA1100
C                     TO BE USED.                                   DGEA1110
C                  METH = 2, IMPLIES THAT THE STIFF METHODS OF      DGEA1120
C                    GEAR, OR THE BACKWARD DIFFERENTIATION          DGEA1130
C                     FORMULAE ARE TO BE USED.                      DGEA1140
C          MITER   - INPUT ITERATION METHOD INDICATOR.              DGEA1150
C                    MITER = 0, IMPLIES THAT FUNCTIONAL             DGEA1160
C                      ITERATION IS USED. NO PARTIAL                DGEA1170
C                      DERIVATIVES ARE NEEDED. A DUMMY FCNJ         DGEA1180
C                       CAN BE USED.                                DGEA1190
C                    MITER = 1, IMPLIES THAT THE CHORD METHOD       DGEA1200
C                      IS USED WITH AN ANALYTIC JACOBIAN. FOR       DGEA1210
C                      THIS METHOD, THE USER SUPPLIES               DGEA1220
C                       SUBROUTINE FCNJ.                            DGEA1230
```

```
C                 MITER = 2, IMPLIES THAT THE CHORD METHOD      DGEA1240
C                   IS USED WITH THE JACOBIAN CALCULATED        DGEA1250
C                   INTERNALLY BY FINITE DIFFERENCES.           DGEA1260
C                   A DUMMY FCNJ CAN BE USED.                   DGEA1270
C                 MITER = 3, IMPLIES THAT THE CHORD METHOD      DGEA1280
C                   IS USED WITH THE JACOBIAN REPLACED BY       DGEA1290
C                   A DIAGONAL APPROXIMATION BASED ON A         DGEA1300
C                   DIRECTIONAL DERIVATIVE.                     DGEA1310
C                   A DUMMY FCNJ CAN BE USED.                   DGEA1320
C                 MITER = -1 OR -2, IMPLIES USE THE SAME        DGEA1330
C                   METHOD AS FOR MITER= 1 OR 2, RESPECTIVELY,DGEA1340
C                   BUT USING A BANDED JACOBIAN MATRIX.  IN     DGEA1350
C                   THESE TWO CASES BANDWIDTH INFORMATION       DGEA1360
C                   MUST BE PASSED TO DGEAR THROUGH THE         DGEA1370
C                   COMMON BLOCK                                DGEA1380
C                      COMMON /DBAND/ NLC,NUC                   DGEA1390
C                   WHERE NLC=NUMBER OF LOWER CODIAGONALS       DGEA1400
C                         NUC=NUMBER OF UPPER CODIAGONALS       DGEA1410
C          INDEX  - INPUT AND OUTPUT PARAMETER USED TO INDICATE DGEA1420
C                 THE TYPE OF CALL TO THE SUBROUTINE.  ON       DGEA1430
C                 OUTPUT INDEX IS RESET TO 0 IF INTEGRATION     DGEA1440
C                 WAS SUCCESSFUL.  OTHERWISE, THE VALUE OF      DGEA1450
C                 INDEX IS UNCHANGED.                           DGEA1460
C               ON INPUT, INDEX = 1, IMPLIES THAT THIS IS THE DGEA1470
C                 FIRST CALL FOR THIS PROBLEM.                  DGEA1480
C               ON INPUT, INDEX = 0, IMPLIES THAT THIS IS NOT DGEA1490
C                 THE FIRST CALL FOR THIS PROBLEM.              DGEA1500
C               ON INPUT, INDEX = -1, IMPLIES THAT THIS IS NOTDGEA1510
C                 THE FIRST CALL FOR THIS PROBLEM, AND THE      DGEA1520
C                 USER HAS RESET TOL.                           DGEA1530
C               ON INPUT, INDEX = 2, IMPLIES THAT THIS IS NOT DGEA1540
C                 THE FIRST CALL FOR THIS PROBLEM. INTEGRATIONDGEA1550
C                 IS TO CONTINUE AND XEND IS TO BE HIT EXACTLYDGEA1560
C                 (NO INTERPOLATION IS DONE). THIS VALUE OF     DGEA1570
C                 INDEX ASSUMES THAT XEND IS BEYOND THE         DGEA1580
C                 CURRENT VALUE OF X.                           DGEA1590
C               ON INPUT, INDEX = 3, IMPLIES THAT THIS IS NOT DGEA1600
C                 THE FIRST CALL FOR THIS PROBLEM. INTEGRATIONDGEA1610
C                 IS TO CONTINUE AND CONTROL IS TO BE RETURNEDDGEA1620
C                 TO THE CALLING PROGRAM AFTER ONE STEP. XEND DGEA1630
C                 IS IGNORED.                                   DGEA1640
C          IWK    - INTEGER WORK VECTOR OF LENGTH N. USED ONLY IF DGEA1650
C                 MITER = 1 OR 2                                DGEA1660
C          WK     - REAL WORK VECTOR OF LENGTH 4*N+NMETH+NMITER.  DGEA1670
C                 THE VALUE OF NMETH DEPENDS ON THE VALUE OF    DGEA1680
C                   METH.                                       DGEA1690
C                   IF METH IS EQUAL TO 1,                      DGEA1700
C                     NMETH IS EQUAL TO N*13.                   DGEA1710
C                   IF METH IS EQUAL TO 2,                      DGEA1720
C                     NMETH IS EQUAL TO N*6.                    DGEA1730
C                 THE VALUE OF NMITER DEPENDS ON THE VALUE OF   DGEA1740
C                   MITER.                                      DGEA1750
C                   IF MITER IS EQUAL TO 1 OR 2,                DGEA1760
C                     NMITER IS EQUAL TO N*(N+1)                DGEA1770
C                   IF MITER IS EQUAL TO -1 OR -2,              DGEA1780
C                     NMITER IS EQUAL TO (2*NLC+NUC+3)*N        DGEA1790
C                       WHERE NLC=NUMBER OF LOWER CODIAGONALS   DGEA1800
C                             NUC=NUMBER OF UPPER CODIAGONALS   DGEA1810
C                   IF MITER IS EQUAL TO 3,                     DGEA1820
C                     NMITER IS EQUAL TO N.                     DGEA1830
C                   IF MITER IS EQUAL TO 0,                     DGEA1840
C                     NMITER IS EQUAL TO 1.                     DGEA1850
```

```
C                        WK MUST REMAIN UNCHANGED BETWEEN SUCCESSIVE DGEA1860
C                         CALLS DURING INTEGRATION.                  DGEA1870
C              IER     - ERROR PARAMETER. (OUTPUT)                   DGEA1880
C                        WARNING ERROR                               DGEA1890
C                         IER = 33, IMPLIES THAT X+H WILL EQUAL X ON DGEA1900
C                           THE NEXT STEP. THIS CONDITION DOES NOT   DGEA1910
C                           FORCE THE ROUTINE TO HALT. HOWEVER, IT   DGEA1920
C                           DOES INDICATE ONE OF TWO CONDITIONS.     DGEA1930
C                           THE USER MIGHT BE REQUIRING TOO MUCH     DGEA1940
C                           ACCURACY VIA THE INPUT PARAMETER TOL.    DGEA1950
C                           IN THIS CASE THE USER SHOULD CONSIDER    DGEA1960
C                           INCREASING THE VALUE OF TOL. THE OTHER   DGEA1970
C                           CONDITION WHICH MIGHT GIVE RISE TO THIS  DGEA1980
C                           ERROR MESSAGE IS THAT THE SYSTEM OF      DGEA1990
C                           DIFFERENTIAL EQUATIONS BEING SOLVED      DGEA2000
C                           IS STIFF (EITHER IN GENERAL OR OVER      DGEA2010
C                           THE SUBINTERVAL OF THE PROBLEM BEING     DGEA2020
C                           SOLVED AT THE TIME OF THE ERROR). IN     DGEA2030
C                           THIS CASE THE USER SHOULD CONSIDER       DGEA2040
C                           USING A NONZERO VALUE FOR THE INPUT      DGEA2050
C                           PARAMETER MITER.                         DGEA2060
C                        WARNING WITH FIX ERROR                      DGEA2070
C                         IER = 66, IMPLIES THAT THE ERROR TEST      DGEA2080
C                           FAILED. H WAS REDUCED BY .1 ONE OR MORE  DGEA2090
C                           TIMES AND THE STEP WAS TRIED AGAIN       DGEA2100
C                           SUCCESSFULLY.                            DGEA2110
C                         IER = 67, IMPLIES THAT CORRECTOR           DGEA2120
C                           CONVERGENCE COULD NOT BE ACHIEVED.       DGEA2130
C                           H WAS REDUCED BY .1 ONE OR MORE TIMES AND DGEA2140
C                           THE STEP WAS TRIED AGAIN SUCCESSFULLY.   DGEA2150
C                        TERMINAL ERROR                              DGEA2160
C                         IER = 132, IMPLIES THE INTEGRATION WAS     DGEA2170
C                           HALTED AFTER FAILING TO PASS THE ERROR   DGEA2180
C                           TEST EVEN AFTER REDUCING H BY A FACTOR   DGEA2190
C                           OF 1.0E10 FROM ITS INITIAL VALUE.        DGEA2200
C                           SEE REMARKS.                             DGEA2210
C                         IER = 133, IMPLIES THE INTEGRATION WAS     DGEA2220
C                           HALTED AFTER FAILING TO ACHIEVE          DGEA2230
C                           CORRECTOR CONVERGENCE EVEN AFTER         DGEA2240
C                           REDUCING H BY A FACTOR OF 1.0E10 FROM    DGEA2250
C                           ITS INITIAL VALUE. SEE REMARKS.          DGEA2260
C                         IER = 134, IMPLIES THAT AFTER SOME INITIAL DGEA2270
C                           SUCCESS, THE INTEGRATION WAS HALTED EITHERDGEA2280
C                           BY REPEATED ERROR TEST FAILURES OR BY    DGEA2290
C                           A TEST ON TOL. SEE REMARKS.              DGEA2300
C                         IER = 135, IMPLIES THAT ONE OF THE INPUT   DGEA2310
C                           PARAMETERS N,X,H,XEND,TOL,METH,MITER, OR DGEA2320
C                           INDEX WAS SPECIFIED INCORRECTLY.         DGEA2330
C                         IER = 136, IMPLIES THAT INDEX HAD A VALUE  DGEA2340
C                           OF -1 ON INPUT, BUT THE DESIRED CHANGES  DGEA2350
C                           OF PARAMETERS WERE NOT IMPLEMENTED       DGEA2360
C                           BECAUSE XEND WAS NOT BEYOND X.           DGEA2370
C                           INTERPOLATION TO X = XEND WAS PERFORMED. DGEA2380
C                           TO TRY AGAIN, SIMPLY CALL AGAIN WITH     DGEA2390
C                           INDEX EQUAL TO -1 AND A NEW VALUE FOR    DGEA2400
C                           XEND.                                    DGEA2410
C                 Step -    # of integration steps taken               +
C                                                                    DGEA2420
C   PRECISION/HARDWARE  - SINGLE AND DOUBLE/H32                      DGEA2430
C                       - SINGLE/H36,H48,H60                         DGEA2440
C                                                                    DGEA2450
C   REQD. IMSL ROUTINES - DGRCS,DGRIN,DGRPS,DGRST,LUDATF,LUELMF,LEQT1B, DGEA2460
```

```
C                       UERTST,UGETIO
C
C  NOTATION           - INFORMATION ON SPECIAL NOTATION AND
C                         CONVENTIONS IS AVAILABLE IN THE MANUAL
C                         INTRODUCTION OR THROUGH IMSL ROUTINE UHELP
C
C  REMARKS  1.  THE EXTERNAL SUBROUTINE FCNJ IS USED ONLY WHEN
C               INPUT PARAMETER MITER IS EQUAL TO 1 OR -1. OTHERWISE,
C               A DUMMY FUNCTION CAN BE USED. THE DUMMY SUBROUTINE
C               SHOULD BE OF THE FOLLOWING FORM
C                 SUBROUTINE FCNJ (N,X,Y,PD)
C                 INTEGER N
C                 REAL Y(N),PD(N,N),X
C                 RETURN
C                 END
C           2.  AFTER THE INITIAL CALL, IF A NORMAL RETURN OCCURRED
C               (IER=0) AND A NORMAL CONTINUATION IS DESIRED, SIMPLY
C               RESET XEND AND CALL DGEAR AGAIN. ALL OTHER
C               PARAMETERS WILL BE READY FOR THE NEXT CALL. A CHANGE
C               OF PARAMETERS WITH INDEX EQUAL TO -1 CAN BE MADE
C               AFTER EITHER A SUCCESSFUL OR AN UNSUCCESSFUL RETURN.
C           3.  THE COMMON BLOCKS /DBAND/ AND /GEAR/ NEED TO BE
C               PRESERVED BETWEEN CALLS TO DGEAR. IF IT IS NECESSARY
C               FOR THE COMMON BLOCKS TO EXIST IN THE CALLING PROGRAM
C               THE FOLLOWING STATEMENTS SHOULD BE INCLUDED
C                 COMMON /DBAND/ NLC,NUC
C                COMMON  /GEAR/ DUMMY(48),SDUMMY(4),IDUMMY(38)
C               WHERE DUMMY, SDUMMY, AND IDUMMY ARE VARIABLE NAMES NOT
C               USED ELSEWHERE IN THE CALLING PROGRAM.  (FOR DOUBLE
C               PRECISION DUMMY IS TYPE DOUBLE AND SDUMMY IS TYPE REAL)
C           4.  THE CHOICE OF VALUES FOR METH AND MITER MAY REQUIRE
C               SOME EXPERIMENTATION, AND ALSO SOME CONSIDERATION OF
C               THE NATURE OF THE PROBLEM AND OF STORAGE REQUIREMENTS.
C               THE PRIME CONSIDERATION IS STIFFNESS. IF
C               THE PROBLEM IS NOT STIFF, THE BEST CHOICE IS PROBABLY
C               METH = 1 WITH MITER = 0. IF THE PROBLEM IS STIFF TO A
C               SIGNIFICANT DEGREE, THEN METH SHOULD BE 2 AND MITER
C               SHOULD BE 1,2,-1,-2 OR 3. IF THE USER HAS NO KNOWLEDGE
C               OF THE INHERENT TIME CONSTANTS OF THE PROBLEM, WITH
C               WHICH TO PREDICT ITS STIFFNESS, ONE WAY TO DETERMINE
C               THIS IS TO TRY METH = 1 AND MITER = 0 FIRST, AND LOOK
C               AT THE BEHAVIOR OF THE SOLUTION COMPUTED AND THE STEP
C               SIZES USED. IF THE TYPICAL VALUES OF H ARE MUCH
C               SMALLER THAN THE SOLUTION BEHAVIOR WOULD SEEM TO
C               REQUIRE (THAT IS, MORE THAN 100 STEPS ARE TAKEN OVER
C               AN INTERVAL IN WHICH THE SOLUTIONS CHANGE BY LESS
C               THAN ONE PERCENT), THEN THE PROBLEM IS PROBABLY STIFF
C               AND THE DEGREE OF STIFFNESS CAN BE ESTIMATED FROM THE
C               VALUES OF H USED AND THE SMOOTHNESS OF THE SOLUTION.
C               IF THE DEGREE OF STIFFNESS IS ONLY SLIGHT, IT MAY BE
C               THAT METH=1 IS MORE EFFICIENT THAN METH=2.
C               EXPERIMENTATION WOULD BE REQUIRED TO DETERMINE THIS.
C               REGARDLESS OF METH, THE LEAST EFFECTIVE VALUE OF
C               MITER IS 0, AND THE MOST EFFECTIVE IS 1,-1,2,OR -2.
C               MITER = 3 IS GENERALLY SOMEWHERE IN BETWEEN. SINCE
C               THE STORAGE REQUIREMENTS GO UP IN THE SAME ORDER AS
C               EFFECTIVENESS, TRADE-OFF CONSIDERATIONS ARE
C               NECESSARY. FOR REASONS OF ACCURACY AND SPEED, THE
C               CHOICE OF ABS(MITER)=1 IS GENERALLY PREFERRED TO
C               ABS(MITER)=2, UNLESS THE SYSTEM IS FAIRLY COMPLICATED
C               (AND FCNJ IS THUS NOT FEASIBLE TO CODE). THE
C               ACCURACY OF THE FCNJ CALCULATION CAN BE CHECKED BY
```

```
C               COMPARISON OF THE JACOBIAN WITH THAT GENERATED WITH
C               ABS(MITER)=2. IF THE JACOBIAN MATRIX IS SIGNIFICANTLY
C               DIAGONALLY DOMINANT, THEN THE OPTION MITER = 3 IS
C               LIKELY TO BE NEARLY AS EFFECTIVE AS ABS(MITER)=1 OR 2,
C               AND WILL SAVE CONSIDERABLE STORAGE AND RUN TIME.
C               IT IS POSSIBLE, AND POTENTIALLY QUITE DESIRABLE, TO
C               USE DIFFERENT VALUES OF METH AND MITER IN DIFFERENT
C               SUBINTERVALS OF THE PROBLEM. FOR EXAMPLE, IF THE
C               PROBLEM IS NON-STIFF INITIALLY AND STIFF LATER,
C               METH = 1 AND MITER = 0 MIGHT BE SET INITIALLY, AND
C               METH = 2 AND MITER = 1 LATER.
C            5. THE INITIAL VALUE OF THE STEP SIZE, H, SHOULD BE
C               CHOSEN CONSIDERABLY SMALLER THAN THE AVERAGE VALUE
C               EXPECTED FOR THE PROBLEM, AS THE FIRST-ORDER METHOD
C               WITH WHICH DGEAR BEGINS IS NOT GENERALLY THE MOST
C               EFFICIENT ONE. HOWEVER, FOR THE FIRST STEP, AS FOR
C               EVERY STEP, DGEAR TESTS FOR THE POSSIBILITY THAT
C               THE STEP SIZE WAS TOO LARGE TO PASS THE ERROR TEST
C               (BASED ON TOL), AND IF SO ADJUSTS THE STEP SIZE
C               DOWN AUTOMATICALLY. THIS DOWNWARD ADJUSTMENT, IF
C               ANY, IS NOTED BY IER HAVING THE VALUES 66 OR 67,
C               AND SUBSEQUENT RUNS ON THE SAME OR SIMILAR PROBLEM
C               SHOULD BE STARTED WITH AN APPROPRIATELY SMALLER
C               VALUE OF H.
C            6. SOME OF THE VALUES OF INTEREST LOCATED IN THE
C               COMMON BLOCK /GEAR/ ARE
C               A. HUSED, THE STEP SIZE H LAST USED SUCCESSFULLY
C                  (DUMMY(8))
C               B. NQUSED, THE ORDER LAST USED SUCCESSFULLY
C                  (IDUMMY(6))
C               C. NSTEP, THE CUMULATIVE NUMBER OF STEPS TAKEN
C                  (IDUMMY(7))
C               D. NFE, THE CUMULATIVE NUMBER OF FCN EVALUATIONS
C                  (IDUMMY(8))
C               E. NJE, THE CUMULATIVE NUMBER OF JACOBIAN
C                  EVALUATIONS, AND HENCE ALSO OF MATRIX LU
C                  DECOMPOSITIONS (IDUMMY(9))
C            7. THE NORMAL USAGE OF DGEAR MAY BE SUMMARIZED AS FOLLOWS
C               A. SET THE INITIAL VALUES IN Y.
C               B. SET N, X, H, TOL, METH, AND MITER.
C               C. SET XEND TO THE FIRST OUTPUT POINT, AND INDEX TO 1.
C               D. CALL DGEAR
C               E. EXIT IF IER IS GREATER THAN 128.
C               F. OTHERWISE, DO DESIRED OUTPUT OF Y.
C               G. EXIT IF THE PROBLEM IS FINISHED.
C               H. OTHERWISE, RESET XEND TO THE NEXT OUTPUT POINT, AND
C                  RETURN TO STEP D.
C            8. THE ERROR WHICH IS CONTROLLED BY WAY OF THE PARAMETER
C               TOL IS AN ESTIMATE OF THE LOCAL TRUNCATION ERROR, THAT
C               IS, THE ERROR COMMITTED ON TAKING A SINGLE STEP WITH
C               THE METHOD, STARTING WITH DATA REGARDED AS EXACT. THIS
C               IS TO BE DISTINGUISHED FROM THE GLOBAL TRUNCATION
C               ERROR, WHICH IS THE ERROR IN ANY GIVEN COMPUTED VALUE
C               OF Y(X) AS A RESULT OF THE LOCAL TRUNCATION ERRORS
C               FROM ALL STEPS TAKEN TO OBTAIN Y(X). THE LATTER ERROR
C               ACCUMULATES IN A NON-TRIVIAL WAY FROM THE LOCAL
C               ERRORS, AND IS NEITHER ESTIMATED NOR CONTROLLED BY
C               THE ROUTINE. SINCE IT IS USUALLY THE GLOBAL ERROR THAT
C               A USER WANTS TO HAVE UNDER CONTROL, SOME
C               EXPERIMENTATION MAY BE NECESSARY TO GET THE RIGHT
C               VALUE OF TOL TO ACHIEVE THE USER'S NEEDS. IF THE
C               PROBLEM IS MATHEMATICALLY STABLE, AND THE METHOD USED
```

```
C                   IS APPROPRIATELY STABLE, THEN THE GLOBAL ERROR AT A
C                   GIVEN X SHOULD VARY SMOOTHLY WITH TOL IN A MONOTONE
C                   INCREASING MANNER.
C                9. IF THE ROUTINE RETURNS WITH IER VALUES OF 132, 133,
C                   OR 134, THE USER SHOULD CHECK TO SEE IF TOO MUCH
C                   ACCURACY IS BEING REQUIRED. THE USER MAY WISH TO
C                   SET TOL TO A LARGER VALUE AND CONTINUE. ANOTHER
C                   POSSIBLE CAUSE OF THESE ERROR CONDITIONS IS AN
C                   ERROR IN THE CODING OF THE EXTERNAL FUNCTIONS FCN
C                   OR FCNJ. IF NO ERRORS ARE FOUND, IT MAY BE NECESSARY
C                   TO MONITOR INTERMEDIATE QUANTITIES GENERATED BY THE
C                   ROUTINE. THESE QUANTITIES ARE STORED IN THE WORK VECTOR
C                   WK AND INDEXED BY SPECIFIC ELEMENTS IN THE COMMON BLOCK
C                   /GEAR/. IF IER IS 132 OR 134, THE COMPONENTS CAUSING
C                   THE ERROR TEST FAILURE CAN BE IDENTIFIED FROM LARGE
C                    VALUES OF THE QUANTITY
C                      WK(IDUMMY(11)+I)/WK(I), FOR I=1,...,N.
C                   ONE CAUSE OF THIS MAY BE A VERY SMALL BUT NONZERO
C                    INITIAL VALUE OF ABS(Y(I)).
C                   IF IER IS 133, SEVERAL POSSIBILITIES EXIST.
C                   IT MAY BE INSTRUCTIVE TO TRY DIFFERENT VALUES OF MITER.
C                   ALTERNATIVELY, THE USER MIGHT MONITOR SUCCESSIVE
C                   CORRECTOR ITERATES CONTAINED IN WK(IDUMMY(12)+I), FOR
C                   I=1,...,N. ANOTHER POSSIBILITY MIGHT BE TO MONITOR
C                   THE JACOBIAN MATRIX, IF ONE IS USED, STORED, BY
C                   COLUMN, IN WK(IDUMMY(10)+I), FOR I=1,...,N*N IF
C                   ABS(MITER) IS EQUAL TO 1 OR 2, OR FOR I=1,...,N IF
C                    MITER IS EQUAL TO 3.
C
C   COPYRIGHT           - 1984 BY IMSL, INC.  ALL RIGHTS RESERVED.
C
C   WARRANTY            - IMSL WARRANTS ONLY THAT IMSL TESTING HAS BEEN
C                           APPLIED TO THIS CODE.  NO OTHER WARRANTY,
C                           EXPRESSED OR IMPLIED, IS APPLICABLE.
C
C-----------------------------------------------------------------------
C
      SUBROUTINE DGEAR  (N,FCN,FCNJ,X,H,Y,XEND,TOL,METH,MITER,INDEX,
     1                   IWK,WK,IER,step)
C                                  SPECIFICATIONS FOR ARGUMENTS
      INTEGER            N,METH,MITER,INDEX,IWK(1),IER,step
      DOUBLE PRECISION   X,H,Y(N),XEND,TOL,WK(1)
C                                  SPECIFICATIONS FOR LOCAL VARIABLES
      INTEGER            NERROR,NSAVE1,NSAVE2,NPW,NY,NC,MFC,KFLAG,
     1                  JSTART,NSQ,NQUSED,NSTEP,NFE,NJE,I,N0,NHCUT,KGO,
     2                   JER,KER,NN,NEQUIL,IDUMMY(21),NLC,NUC
      REAL               SDUMMY(4)
      DOUBLE PRECISION   T,HH,HMIN,HMAX,EPSC,UROUND,EPSJ,HUSED,TOUTP,
     1                   AYI,D,DN,SEPS,DUMMY(39)
      EXTERNAL           FCN,FCNJ
      COMMON /DBAND/     NLC,NUC
      COMMON /GEAR/      T,HH,HMIN,HMAX,EPSC,UROUND,EPSJ,HUSED,DUMMY,
     1                  TOUTP,SDUMMY,NC,MFC,KFLAG,JSTART,NSQ,NQUSED,
     2                  NSTEP,NFE,NJE,NPW,NERROR,NSAVE1,NSAVE2,NEQUIL,
     3                   NY,IDUMMY,N0,NHCUT
      DATA               SEPS/Z3410000000000000/
C                                  FIRST EXECUTABLE STATEMENT
      IF (MITER.GE.0) NLC = -1
      KER = 0
      JER = 0
      UROUND = SEPS
C                                  COMPUTE WORK VECTOR INDICIES
```

```
      NERROR = N
      NSAVE1 = NERROR+N
      NSAVE2 = NSAVE1+N
      NY = NSAVE2+N
      IF (METH.EQ.1) NEQUIL = NY+13*N
      IF (METH.EQ.2) NEQUIL = NY+6*N
      NPW = NEQUIL + N
      IF (MITER.EQ.0.OR.MITER.EQ.3) NPW = NEQUIL
      MFC = 10*METH+IABS(MITER)
C                                   CHECK FOR INCORRECT INPUT PARAMETERS
C
      IF (MITER.LT.-2.OR.MITER.GT.3) GO TO 85
      IF (METH.NE.1.AND.METH.NE.2) GO TO 85
      IF (TOL.LE.0.D0) GO TO 85
      IF (N.LE.0) GO TO 85
      IF ((X-XEND)*H.GE.0.D0) GO TO 85
      IF (INDEX.EQ.0) GO TO 10
      IF (INDEX.EQ.2) GO TO 15
      IF (INDEX.EQ.-1) GO TO 20
      IF (INDEX.EQ.3) GO TO 25
      IF (INDEX.NE.1) GO TO 85
C                                   IF INITIAL VALUES OF YMAX OTHER THAN
C                                     THOSE SET BELOW ARE DESIRED, THEY
C                                     SHOULD BE SET HERE. ALL YMAX(I)
C                                     MUST BE POSITIVE. IF VALUES FOR
C                                     HMIN OR HMAX, THE BOUNDS ON
C                                     DABS(HH), OTHER THAN THOSE BELOW
C                                     ARE DESIRED, THEY SHOULD BE SET
C                                     BELOW.
      DO 5 I=1,N
         WK(I) = DABS(Y(I))
         IF (WK(I).EQ.0.D0) WK(I) = 1.D0
         WK(NY+I) = Y(I)
    5 CONTINUE
      NC = N
      T = X
      HH = H
      IF ((T+HH).EQ.T) KER = 33
      HMIN = DABS(H)
      HMAX = DABS(X-XEND)*10.D0
      EPSC = TOL
      JSTART = 0
      N0 = N
      NSQ = N0*N0
      EPSJ = DSQRT(UROUND)
      NHCUT = 0
      DUMMY(2) = 1.0D0
      DUMMY(14) = 1.0D0
      GO TO 30
C                                   TOUTP IS THE PREVIOUS VALUE OF XEND
C                                     FOR USE IN HMAX.
   10 HMAX = DABS(XEND-TOUTP)*10.D0
      GO TO 45
C
   15 HMAX = DABS(XEND-TOUTP)*10.D0
      IF ((T-XEND)*HH.GE.0.D0) GO TO 95
      GO TO 50
C
   20 IF ((T-XEND)*HH.GE.0.D0) GO TO 90
      JSTART = -1
      NC = N
      EPSC = TOL
```

```
C                                                                      DGEA4950
   25 IF ((T+HH).EQ.T) KER = 33                                        DGEA4960
      write(*,*),'error code = ',ker                                   +
C                                                                      DGEA4970
   30 NN = N0                                                          DGEA4980
      step = step + 1                                                  +
      write(*,*)'step = ',step                                         +
      CALL DGRST (FCN,FCNJ,WK(NY+1),WK,WK(NERROR+1),WK(NSAVE1+1),      DGEA4990
     1 WK(NSAVE2+1),WK(NPW+1),WK(NEQUIL+1),IWK,NN,step)                +
C                                                                      DGEA5010
      KGO = 1-KFLAG                                                    DGEA5020
      GO TO (35,55,70,80), KGO                                         DGEA5030
C                                  KFLAG = 0, -1, -2, -3               DGEA5040
   35 CONTINUE                                                         DGEA5050
C                                  NORMAL RETURN FROM INTEGRATOR. THE  DGEA5060
C                                    WEIGHTS YMAX(I) ARE UPDATED. IF   DGEA5070
C                                    DIFFERENT VALUES ARE DESIRED, THEY DGEA5080
C                                    SHOULD BE SET HERE. A TEST IS MADE DGEA5090
C                                    FOR TOL BEING TOO SMALL FOR THE   DGEA5100
C                                    MACHINE PRECISION. ANY OTHER TESTS DGEA5110
C                                    OR CALCULATIONS THAT ARE REQUIRED DGEA5120
C                                    AFTER EVERY STEP SHOULD BE        DGEA5130
C                                    INSERTED HERE. IF INDEX = 3, Y IS DGEA5140
C                                    SET TO THE CURRENT SOLUTION ON    DGEA5150
C                                    RETURN. IF INDEX = 2, HH IS       DGEA5160
C                                    CONTROLLED TO HIT XEND (WITHIN    DGEA5170
C                                    ROUNDOFF ERROR), AND THEN THE     DGEA5180
C                                    CURRENT SOLUTION IS PUT IN Y ON   DGEA5190
C                                    RETURN. FOR ANY OTHER VALUE OF    DGEA5200
C                                    INDEX, CONTROL RETURNS TO THE     DGEA5210
C                                    INTEGRATOR UNLESS XEND HAS BEEN   DGEA5220
C                                    REACHED. THEN INTERPOLATED VALUES DGEA5230
C                                    OF THE SOLUTION ARE COMPUTED AND  DGEA5240
C                                    STORED IN Y ON RETURN.            DGEA5250
C                                    IF INTERPOLATION IS NOT           DGEA5260
C                                    DESIRED, THE CALL TO DGRIN SHOULD DGEA5270
C                                    BE REMOVED AND CONTROL TRANSFERRED DGEA5280
C                                    TO STATEMENT 95 INSTEAD OF 105.   DGEA5290
      D = 0.D0                                                         DGEA5300
      DO 40 I=1,N                                                      DGEA5310
         AYI = DABS(WK(NY+I))                                          DGEA5320
         WK(I) = DMAX1(WK(I),AYI)                                      DGEA5330
   40 D = D+(AYI/WK(I))**2                                             DGEA5340
      D = D*(UROUND/TOL)**2                                            DGEA5350
      DN = N                                                           DGEA5360
      IF (D.GT.DN) GO TO 75                                            DGEA5370
      IF (INDEX.EQ.3) GO TO 95                                         DGEA5380
      IF (INDEX.EQ.2) GO TO 50                                         DGEA5390
   45 IF ((T-XEND)*HH.LT.0.D0) GO TO 25                                DGEA5400
      NN = N0                                                          DGEA5410
      CALL DGRIN (XEND,WK(NY+1),NN,Y)                                  DGEA5420
      X = XEND                                                         DGEA5430
      GO TO 105                                                        DGEA5440
   50 IF (((T+HH)-XEND)*HH.LE.0.D0) GO TO 25                           DGEA5450
      IF (DABS(T-XEND).LE.UROUND*DMAX1(10.D0*DABS(T),HMAX)) GO TO 95   DGEA5460
      IF ((T-XEND)*HH.GE.0.D0) GO TO 95                                DGEA5470
      HH = (XEND-T)*(1.D0-4.D0*UROUND)                                 DGEA5480
      JSTART = -1                                                      DGEA5490
      GO TO 25                                                         DGEA5500
C                                  ON AN ERROR RETURN FROM INTEGRATOR, DGEA5510
C                                    AN IMMEDIATE RETURN OCCURS IF     DGEA5520
C                                    KFLAG = -2, AND RECOVERY ATTEMPTS DGEA5530
```

```
C                                  ARE MADE OTHERWISE. TO RECOVER, HH DGEA5540
C                                  AND HMIN ARE REDUCED BY A FACTOR   DGEA5550
C                                  OF .1 UP TO 10 TIMES BEFORE GIVING DGEA5560
C                                  UP.                                DGEA5570
   55 JER = 66                                                        DGEA5580
   60 IF (NHCUT.EQ.10) GO TO 65                                       DGEA5590
      NHCUT = NHCUT+1                                                 DGEA5600
      HMIN = HMIN*.1D0                                                DGEA5610
      HH = HH*.1D0                                                    DGEA5620
      JSTART = -1                                                     DGEA5630
      GO TO 25                                                        DGEA5640
C                                                                     DGEA5650
   65 IF (JER.EQ.66) JER = 132                                        DGEA5660
      IF (JER.EQ.67) JER = 133                                        DGEA5670
      GO TO 95                                                        DGEA5680
C                                                                     DGEA5690
   70 JER = 134                                                       DGEA5700
      GO TO 95                                                        DGEA5710
C                                                                     DGEA5720
   75 JER = 134                                                       DGEA5730
      KFLAG = -2                                                      DGEA5740
      GO TO 95                                                        DGEA5750
C                                                                     DGEA5760
   80 JER = 67                                                        DGEA5770
      GO TO 60                                                        DGEA5780
C                                                                     DGEA5790
   85 JER = 135                                                       DGEA5800
      GO TO 110                                                       DGEA5810
C                                                                     DGEA5820
   90 JER = 136                                                       DGEA5830
      NN = N0                                                         DGEA5840
      CALL DGRIN (XEND,WK(NY+1),NN,Y)                                 DGEA5850
      X = XEND                                                        DGEA5860
      GO TO 110                                                       DGEA5870
C                                                                     DGEA5880
   95 X = T                                                           DGEA5890
      DO 100 I=1,N                                                    DGEA5900
  100 Y(I) = WK(NY+I)                                                 DGEA5910
  105 IF (JER.LT.128) INDEX = KFLAG                                   DGEA5920
      TOUTP = X                                                       DGEA5930
      IF (KFLAG.EQ.0) H = HUSED                                       DGEA5940
      IF (KFLAG.NE.0) H = HH                                          DGEA5950
  110 IER = MAX0(KER,JER)                                             DGEA5960
 9000 CONTINUE                                                        DGEA5970
      IF (KER.NE.0.AND.JER.LT.128) CALL UERTST (KER,6HDGEAR )         DGEA5980
      IF (JER.NE.0) CALL UERTST (JER,6HDGEAR )                        DGEA5990
 9005 RETURN                                                          DGEA6000
      END                                                             DGEA6010
```

```
C   IMSL ROUTINE NAME   - DGRST                                          DGRS0010
C                                                                        DGRS0020
C-modified to print sheath and diagnostic output to files "sheatha.dat" +
C and "diag.dat"                                                         +
C   COMPUTER            - IBM/DOUBLE                                     DGRS0050
C                                                                        DGRS0060
C   LATEST REVISION     - JUNE 1, 1982                                   DGRS0070
C                                                                        DGRS0080
C   PURPOSE             - NUCLEUS CALLED ONLY BY IMSL SUBROUTINE DGEAR   DGRS0090
C                                                                        DGRS0100
C   PRECISION/HARDWARE  - SINGLE AND DOUBLE/H32                          DGRS0110
C                       - SINGLE/H36,H48,H60                             DGRS0120
C                                                                        DGRS0130
C   REQD. IMSL ROUTINES - DGRCS,DGRPS,LUDATF,LUELMF,LEQT1B,UERTST,       DGRS0140
C                           UGETIO                                       DGRS0150
C                                                                        DGRS0160
C   NOTATION            - INFORMATION ON SPECIAL NOTATION AND            DGRS0170
C                           CONVENTIONS IS AVAILABLE IN THE MANUAL       DGRS0180
C                           INTRODUCTION OR THROUGH IMSL ROUTINE UHELP   DGRS0190
C                                                                        DGRS0200
C   COPYRIGHT           - 1982 BY IMSL, INC. ALL RIGHTS RESERVED.        DGRS0210
C                                                                        DGRS0220
C   WARRANTY            - IMSL WARRANTS ONLY THAT IMSL TESTING HAS BEEN  DGRS0230
C                           APPLIED TO THIS CODE. NO OTHER WARRANTY,     DGRS0240
C                           EXPRESSED OR IMPLIED, IS APPLICABLE.         DGRS0250
C                                                                        DGRS0260
C-----------------------------------------------------------------------DGRS0270
C                                                                        DGRS0280
      SUBROUTINE DGRST  (FCN,FCNJ,Y,YMAX,ERROR,SAVE1,SAVE2,PW,EQUIL,     DGRS0290
     1                   IPIV,N0,step)                                   +
C                                  SPECIFICATIONS FOR ARGUMENTS          DGRS0310
      INTEGER            IPIV(1),N0                                      DGRS0320
      DOUBLE PRECISION   Y(N0,1),YMAX(1),ERROR(1),SAVE1(1),SAVE2(1),     DGRS0330
     1                   PW(1),EQUIL(1),eprime,eprime(2)                 +
C                                  SPECIFICATIONS FOR LOCAL VARIABLES    DGRS0350
      INTEGER            N,MF,KFLAG,JSTART,NQUSED,NSTEP,NFE,NJE,NSQ,     DGRS0360
     1                   I,METH,MITER,NQ,L,IDOUB,MFOLD,NOLD,IRET,MEO,    DGRS0370
     2                   MIO,IWEVAL,MAXDER,LMAX,IREDO,J,NSTEPJ,J1,J2,    DGRS0380
     3                   M,IER,NEWQ,NPW,NERROR,NSAVE1,NSAVE2,NEQUIL,NY,  DGRS0390
     4                   MITER1,IDUMMY(2),NLC,NUC,NWK,JER                DGRS0400
      REAL               TQ(4)                                           DGRS0410
      DOUBLE PRECISION   T,H,HMIN,HMAX,EPS,UROUND,HUSED,EL(13),OLDL0,    DGRS0420
     1                   TOLD,RMAX,RC,CRATE,EPSOLD,HOLD,FN,EDN,E,EUP,    DGRS0430
     2                   BND,RH,R1,CON,R,HL0,R0,D,PHL0,PR3,D1,ENQ3,ENQ2,DGRS0440
     3                   PR2,PR1,ENQ1,EPSJ,DUMMY,tcum                    +
      EXTERNAL           FCN,FCNJ                                        DGRS0460
      COMMON /DBAND/     NLC,NUC                                         DGRS0470
      COMMON /GEAR/      T,H,HMIN,HMAX,EPS,UROUND,EPSJ,HUSED,            DGRS0480
     1                   EL,OLDL0,TOLD,RMAX,RC,CRATE,EPSOLD,HOLD,FN,     DGRS0490
     2                   EDN,E,EUP,BND,RH,R1,R,HL0,R0,D,PHL0,PR3,D1,     DGRS0500
     3                   ENQ3,ENQ2,PR2,PR1,ENQ1,DUMMY,TQ,                DGRS0510
     4                   N,MF,KFLAG,JSTART,NSQ,NQUSED,NSTEP,NFE,NJE,     DGRS0520
     5                   NPW,NERROR,NSAVE1,NSAVE2,NEQUIL,NY,             DGRS0530
     6                   I,METH,MITER,NQ,L,IDOUB,MFOLD,NOLD,IRET,MEO,    DGRS0540
     7                   MIO,IWEVAL,MAXDER,LMAX,IREDO,J,NSTEPJ,J1,J2,    DGRS0550
     8                   M,NEWQ,IDUMMY                                   DGRS0560
C                                  FIRST EXECUTABLE STATEMENT            DGRS0570
      open(unit=8,file='sheatha.dat',status='unknown')                   +
      open(unit=9,file='diag.dat',status='unknown')                      +
      KFLAG = 0                                                          DGRS0580
      TOLD = T                                                           DGRS0590
C                                  THIS ROUTINE PERFORMS ONE STEP OF     DGRS0600
```

```
C                                    THE INTEGRATION OF AN INITIAL
C                                    VALUE PROBLEM FOR A SYSTEM OF
C                                    ORDINARY DIFFERENTIAL EQUATIONS.
      IF (JSTART.GT.0) GO TO 50
      IF (JSTART.NE.0) GO TO 10
C                                  ON THE FIRST CALL, THE ORDER IS SET
C                                    TO 1 AND THE INITIAL YDOT IS
C                                    CALCULATED. RMAX IS THE MAXIMUM
C                                    RATIO BY WHICH H CAN BE INCREASED
C                                    IN A SINGLE STEP. IT IS INITIALLY
C                                    1.E4 TO COMPENSATE FOR THE SMALL
C                                    INITIAL H, BUT THEN IS NORMALLY
C                                    EQUAL TO 10. IF A FAILURE OCCURS
C                                    (IN CORRECTOR CONVERGENCE OR ERROR
C                                    TEST), RMAX IS SET AT 2 FOR THE
C                                    NEXT INCREASE.
      CALL FCN (N,T,Y,SAVE1,eprime,eprime2)
      DO 5 I=1,N
    5 Y(I,2) = H*SAVE1(I)
      METH = MF/10
      MITER = MF-10*METH
      NQ = 1
      L = 2
      IDOUB = 3
      RMAX = 1.D4
      RC = 0.D0
      CRATE = 1.D0
      HOLD = H
      MFOLD = MF
      NSTEP = 0
      NSTEPJ = 0
      NFE = 1
      NJE = 0
      IRET = 3
      GO TO 15
C                                  IF THE CALLER HAS CHANGED METH,
C                                    DGRCS IS CALLED TO SET THE
C                                    COEFFICIENTS OF THE METHOD. IF THE
C                                    CALLER HAS CHANGED N, EPS, OR
C                                    METH, THE CONSTANTS E, EDN, EUP,
C                                    AND BND MUST BE RESET. E IS A
C                                    COMPARISON FOR ERRORS OF THE
C                                    CURRENT ORDER NQ. EUP IS TO TEST
C                                    FOR INCREASING THE ORDER, EDN FOR
C                                    DECREASING THE ORDER. BND IS USED
C                                    TO TEST FOR CONVERGENCE OF THE
C                                    CORRECTOR ITERATES. IF THE CALLER
C                                    HAS CHANGED H, Y MUST BE RESCALED.
C                                    IF H OR METH HAS BEEN CHANGED,
C                                    IDOUB IS RESET TO L + 1 TO PREVENT
C                                    FURTHER CHANGES IN H FOR THAT MANY
C                                    STEPS.
   10 IF (MF.EQ.MFOLD) GO TO 25
      MEO = METH
      MIO = MITER
      METH = MF/10
      MITER = MF-10*METH
      MFOLD = MF
      IF (MITER.NE.MIO) IWEVAL = MITER
      IF (METH.EQ.MEO) GO TO 25
      IDOUB = L+1
      IRET = 1
```

```
   15 CALL DGRCS (METH,NQ,EL,TQ,MAXDER)
      LMAX = MAXDER+1
      RC = RC*EL(1)/OLDL0
      OLDL0 = EL(1)
   20 FN = N
      EDN = FN*(TQ(1)*EPS)**2
      E = FN*(TQ(2)*EPS)**2
      EUP = FN*(TQ(3)*EPS)**2
      BND = FN*(TQ(4)*EPS)**2
      EPSOLD = EPS
      NOLD = N
      GO TO (30,35,50), IRET
   25 IF ((EPS.EQ.EPSOLD).AND.(N.EQ.NOLD)) GO TO 30
      IF (N.EQ.NOLD) IWEVAL = MITER
      IRET = 1
      GO TO 20
   30 IF (H.EQ.HOLD) GO TO 50
      RH = H/HOLD
      H = HOLD
      IREDO = 3
      GO TO 40
   35 RH = DMAX1(RH,HMIN/DABS(H))
   40 RH = DMIN1(RH,HMAX/DABS(H),RMAX)
      R1 = 1.D0
      DO 45 J=2,L
        R1 = R1*RH
      DO 45 I=1,N
   45 Y(I,J) = Y(I,J)*R1
      H = H*RH
      RC = RC*RH
      IDOUB = L+1
      IF (IREDO.EQ.0) GO TO 285
C                                THIS SECTION COMPUTES THE PREDICTED
C                                  VALUES BY EFFECTIVELY MULTIPLYING
C                                  THE Y ARRAY BY THE PASCAL TRIANGLE
C                                  MATRIX. RC IS THE RATIO OF NEW TO
C                                  OLD VALUES OF THE COEFFICIENT
C                                  H*EL(1). WHEN RC DIFFERS FROM 1 BY
C                                  MORE THAN 30 PERCENT, OR THE
C                                  CALLER HAS CHANGED MITER, IWEVAL
C                                  IS SET TO MITER TO FORCE THE
C                                  PARTIALS TO BE UPDATED, IF
C                                  PARTIALS ARE USED. IN ANY CASE,
C                                  THE PARTIALS ARE UPDATED AT LEAST
C                                  EVERY 20-TH STEP.
   50 IF (DABS(RC-1.D0).GT.0.3D0) IWEVAL = MITER
      IF (NSTEP.GE.NSTEPJ+20) IWEVAL = MITER
      T = T+H
      DO 55 J1=1,NQ
      DO 55 J2=J1,NQ
         J = (NQ+J1)-J2
      DO 55 I=1,N
   55 Y(I,J) = Y(I,J)+Y(I,J+1)
C                                UP TO 3 CORRECTOR ITERATIONS ARE
C                                  TAKEN. A CONVERGENCE TEST IS MADE
C                                  ON THE R.M.S. NORM OF EACH
C                                  CORRECTION, USING BND, WHICH IS
C                                  DEPENDENT ON EPS. THE SUM OF THE
C                                  CORRECTIONS IS ACCUMULATED IN THE
C                                  VECTOR ERROR(I). THE Y ARRAY IS
C                                  NOT ALTERED IN THE CORRECTOR LOOP.
C                                  THE UPDATED Y VECTOR IS STORED
```

```
C                                        TEMPORARILY IN SAVE1.          DGRS1850
   60 DO 65 I=1,N                                                       DGRS1860
   65 ERROR(I) = 0.D0                                                   DGRS1870
      M = 0                                                             DGRS1880
      CALL FCN (N,T,Y,SAVE2,eprime,eprime2)                                 +
      NFE = NFE+1                                                       DGRS1900
      IF (IWEVAL.LE.0) GO TO 95                                         DGRS1910
C                                      IF INDICATED, THE MATRIX P = I - DGRS1920
C                                        H*EL(1)*J IS REEVALUATED BEFORE DGRS1930
C                                        STARTING THE CORRECTOR ITERATION. DGRS1940
C                                        IWEVAL IS SET TO 0 AS AN INDICATOR DGRS1950
C                                        THAT THIS HAS BEEN DONE. IF MITER DGRS1960
C                                        = 1 OR 2, P IS COMPUTED AND    DGRS1970
C                                        PROCESSED IN PSET. IF MITER = 3, DGRS1980
C                                        THE MATRIX USED IS P = I -     DGRS1990
C                                        H*EL(1)*D, WHERE D IS A DIAGONAL DGRS2000
C                                        MATRIX.                        DGRS2010
      IWEVAL = 0                                                        DGRS2020
      RC = 1.D0                                                         DGRS2030
      NJE = NJE+1                                                       DGRS2040
      NSTEPJ = NSTEP                                                    DGRS2050
      GO TO (75,70,80), MITER                                           DGRS2060
   70 NFE = NFE+N                                                       DGRS2070
   75 CON = -H*EL(1)                                                    DGRS2080
      MITER1 = MITER                                                    DGRS2090
      CALL DGRPS (FCN,FCNJ,Y,N0,CON,MITER1,YMAX,SAVE1,SAVE2,PW,EQUIL,   DGRS2100
     1 IPIV,IER)                                                        DGRS2110
      IF (IER.NE.0) GO TO 155                                           DGRS2120
      GO TO 125                                                         DGRS2130
   80 R = EL(1)*.1D0                                                    DGRS2140
      DO 85 I=1,N                                                       DGRS2150
   85 PW(I) = Y(I,1)+R*(H*SAVE2(I)-Y(I,2))                              DGRS2160
      CALL FCN (N,T,PW,SAVE1,eprime,eprime2)                                +
      NFE = NFE+1                                                       DGRS2180
      HL0 = H*EL(1)                                                     DGRS2190
      DO 90 I=1,N                                                       DGRS2200
         R0 = H*SAVE2(I)-Y(I,2)                                         DGRS2210
         PW(I) = 1.D0                                                   DGRS2220
         D = .1D0*R0-H*(SAVE1(I)-SAVE2(I))                              DGRS2230
         SAVE1(I) = 0.D0                                                DGRS2240
         IF (DABS(R0).LT.UROUND*YMAX(I)) GO TO 90                       DGRS2250
         IF (DABS(D).EQ.0.D0) GO TO 155                                 DGRS2260
         PW(I) = .1D0*R0/D                                              DGRS2270
         SAVE1(I) = PW(I)*R0                                            DGRS2280
   90 CONTINUE                                                          DGRS2290
      GO TO 135                                                         DGRS2300
   95 IF (MITER.NE.0) GO TO (125,125,105), MITER                        DGRS2310
C                                                                       DGRS2320
C                                      IN THE CASE OF FUNCTIONAL ITERATION, DGRS2330
C                                        UPDATE Y DIRECTLY FROM THE RESULT DGRS2340
C                                        OF THE LAST FCN CALL.          DGRS2350
      D = 0.D0                                                          DGRS2360
      DO 100 I=1,N                                                      DGRS2370
         R = H*SAVE2(I)-Y(I,2)                                          DGRS2380
         D = D+((R-ERROR(I))/YMAX(I))**2                                DGRS2390
         SAVE1(I) = Y(I,1)+EL(1)*R                                      DGRS2400
  100 ERROR(I) = R                                                      DGRS2410
      GO TO 145                                                         DGRS2420
C                                      IN THE CASE OF THE CHORD METHOD, DGRS2430
C                                        COMPUTE THE CORRECTOR ERROR, F SUB DGRS2440
C                                        (M), AND SOLVE THE LINEAR SYSTEM DGRS2450
C                                        WITH THAT AS RIGHT-HAND SIDE AND P DGRS2460
```

```
C                                           AS COEFFICIENT MATRIX, USING THE
C                                           LU DECOMPOSITION IF MITER = 1 OR
C                                           2. IF MITER = 3, THE COEFFICIENT
C                                           H*EL(1) IN P IS UPDATED.
  105 PHL0 = HL0
      HL0 = H*EL(1)
      IF (HL0.EQ.PHL0) GO TO 115
      R = HL0/PHL0
      DO 110 I=1,N
         D = 1.D0-R*(1.D0-1.D0/PW(I))
         IF (DABS(D).EQ.0.D0) GO TO 165
  110 PW(I) = 1.D0/D
  115 DO 120 I=1,N
  120 SAVE1(I) = PW(I)*(H*SAVE2(I)-(Y(I,2)+ERROR(I)))
      GO TO 135
  125 DO 130 I=1,N
  130 SAVE1(I) = H*SAVE2(I)-(Y(I,2)+ERROR(I))
      IF (NLC .EQ. -1) GO TO 131
      NWK = (NLC+NUC+1)*N0+1
      CALL LEQT1B(PW,N,NLC,NUC,N0,SAVE1,1,N0,2,PW(NWK),JER)
      GO TO 135
  131 CALL LUELMF (PW,SAVE1,IPIV,N,N0,SAVE1)
  135 D = 0.D0
      DO 140 I=1,N
         ERROR(I) = ERROR(I)+SAVE1(I)
         D = D+(SAVE1(I)/YMAX(I))**2
  140 SAVE1(I) = Y(I,1)+EL(1)*ERROR(I)
C                                         TEST FOR CONVERGENCE. IF M.GT.0, THE
C                                           SQUARE OF THE CONVERGENCE RATE
C                                           CONSTANT IS ESTIMATED AS CRATE,
C                                           AND THIS IS USED IN THE TEST.
  145 IF (M.NE.0) CRATE = DMAX1(.9D0*CRATE,D/D1)
      IF ((D*DMIN1(1.D0,2.D0*CRATE)).LE.BND) GO TO 170
      D1 = D
      M = M+1
      IF (M.EQ.3) GO TO 150
      CALL FCN (N,T,SAVE1,SAVE2,eprime,eprime2)
      GO TO 95
C                                         THE CORRECTOR ITERATION FAILED TO
C                                           CONVERGE IN 3 TRIES. IF PARTIALS
C                                           ARE INVOLVED BUT ARE NOT UP TO
C                                           DATE, THEY ARE REEVALUATED FOR THE
C                                           NEXT TRY. OTHERWISE THE Y ARRAY IS
C                                           RETRACTED TO ITS VALUES BEFORE
C                                           PREDICTION, AND H IS REDUCED, IF
C                                           POSSIBLE. IF NOT, A NO-CONVERGENCE
C                                           EXIT IS TAKEN.
  150 NFE = NFE+2
      IF (IWEVAL.EQ.-1) GO TO 165
  155 T = TOLD
      RMAX = 2.D0
      DO 160 J1=1,NQ
      DO 160 J2=J1,NQ
         J = (NQ+J1)-J2
      DO 160 I=1,N
  160 Y(I,J) = Y(I,J)-Y(I,J+1)
      IF (DABS(H).LE.HMIN*1.00001D0) GO TO 280
      RH = .25D0
      IREDO = 1
      GO TO 35
  165 IWEVAL = MITER
      GO TO 60
```

```
C                                 THE CORRECTOR HAS CONVERGED. IWEVAL
C                                   IS SET TO -1 IF PARTIAL
C                                   DERIVATIVES WERE USED, TO SIGNAL
C                                   THAT THEY MAY NEED UPDATING ON
C                                   SUBSEQUENT STEPS. THE ERROR TEST
C                                   IS MADE AND CONTROL PASSES TO
C                                   STATEMENT 190 IF IT FAILS.
  170 IF (MITER.NE.0) IWEVAL = -1
      NFE = NFE+M
      D = 0.D0
      DO 175 I=1,N
  175 D = D+(ERROR(I)/YMAX(I))**2
      IF (D.GT.E) GO TO 190
C                                 AFTER A SUCCESSFUL STEP, UPDATE THE
C                                   Y ARRAY. CONSIDER CHANGING H IF
C                                   IDOUB = 1. OTHERWISE DECREASE
C                                   IDOUB BY 1. IF IDOUB IS THEN 1 AND
C                                   NQ .LT. MAXDER, THEN ERROR IS
C                                   SAVED FOR USE IN A POSSIBLE ORDER
C                                   INCREASE ON THE NEXT STEP. IF A
C                                   CHANGE IN H IS CONSIDERED, AN
C                                   INCREASE OR DECREASE IN ORDER BY
C                                   ONE IS CONSIDERED ALSO. A CHANGE
C                                   IN H IS MADE ONLY IF IT IS BY A
C                                   FACTOR OF AT LEAST 1.1. IF NOT,
C                                   IDOUB IS SET TO 10 TO PREVENT
C                                   TESTING FOR THAT MANY STEPS.
      KFLAG = 0
      IREDO = 0
      NSTEP = NSTEP+1
      HUSED = H
      NQUSED = NQ
      DO 180 J=1,L
      DO 180 I=1,N
  180 Y(I,J) = Y(I,J)+EL(J)*ERROR(I)
      IF (IDOUB.EQ.1) GO TO 200
      IDOUB = IDOUB-1
      IF (IDOUB.GT.1) GO TO 290
      IF (L.EQ.LMAX) GO TO 290
      DO 185 I=1,N
  185 Y(I,LMAX) = ERROR(I)
      GO TO 290
C                                 THE ERROR TEST FAILED. KFLAG KEEPS
C                                   TRACK OF MULTIPLE FAILURES.
C                                   RESTORE T AND THE Y ARRAY TO THEIR
C                                   PREVIOUS VALUES, AND PREPARE TO
C                                   TRY THE STEP AGAIN. COMPUTE THE
C                                   OPTIMUM STEP SIZE FOR THIS OR ONE
C                                   LOWER ORDER.
  190 KFLAG = KFLAG-1
      T = TOLD
      DO 195 J1=1,NQ
      DO 195 J2=J1,NQ
         J = (NQ+J1)-J2
      DO 195 I=1,N
  195 Y(I,J) = Y(I,J)-Y(I,J+1)
      RMAX = 2.D0
      IF (DABS(H).LE.HMIN*1.00001D0) GO TO 270
      IF (KFLAG.LE.-3) GO TO 260
      IREDO = 2
      PR3 = 1.D+20
      GO TO 210
```

```
C                                         REGARDLESS OF THE SUCCESS OR FAILURE DGRS3710
C                                           OF THE STEP, FACTORS PR1, PR2, AND DGRS3720
C                                           PR3 ARE COMPUTED, BY WHICH H COULD DGRS3730
C                                           BE DIVIDED AT ORDER NQ - 1, ORDER  DGRS3740
C                                           NQ, OR ORDER NQ + 1, RESPECTIVELY. DGRS3750
C                                           IN THE CASE OF FAILURE, PR3 =      DGRS3760
C                                           1.E20 TO AVOID AN ORDER INCREASE.  DGRS3770
C                                           THE SMALLEST OF THESE IS           DGRS3780
C                                           DETERMINED AND THE NEW ORDER       DGRS3790
C                                           CHOSEN ACCORDINGLY. IF THE ORDER   DGRS3800
C                                           IS TO BE INCREASED, WE COMPUTE ONE DGRS3810
C                                           ADDITIONAL SCALED DERIVATIVE.      DGRS3820
  200 PR3 = 1.D+20                                                       DGRS3830
      IF (L.EQ.LMAX) GO TO 210                                           DGRS3840
      D1 = 0.D0                                                          DGRS3850
      DO 205 I=1,N                                                       DGRS3860
  205 D1 = D1+((ERROR(I)-Y(I,LMAX))/YMAX(I))**2                          DGRS3870
      ENQ3 = .5D0/(L+1)                                                  DGRS3880
      PR3 = ((D1/EUP)**ENQ3)*1.4D0+1.4D-6                                DGRS3890
  210 ENQ2 = .5D0/L                                                      DGRS3900
      PR2 = ((D/E)**ENQ2)*1.2D0+1.2D-6                                   DGRS3910
      PR1 = 1.D+20                                                       DGRS3920
      IF (NQ.EQ.1) GO TO 220                                             DGRS3930
      D = 0.D0                                                           DGRS3940
      DO 215 I=1,N                                                       DGRS3950
  215 D = D+(Y(I,L)/YMAX(I))**2                                          DGRS3960
      ENQ1 = .5D0/NQ                                                     DGRS3970
      PR1 = ((D/EDN)**ENQ1)*1.3D0+1.3D-6                                 DGRS3980
  220 IF (PR2.LE.PR3) GO TO 225                                          DGRS3990
      IF (PR3.LT.PR1) GO TO 235                                          DGRS4000
      GO TO 230                                                          DGRS4010
  225 IF (PR2.GT.PR1) GO TO 230                                          DGRS4020
      NEWQ = NQ                                                          DGRS4030
      RH = 1.D0/PR2                                                      DGRS4040
      GO TO 250                                                          DGRS4050
  230 NEWQ = NQ-1                                                        DGRS4060
      RH = 1.D0/PR1                                                      DGRS4070
      IF (KFLAG.NE.0.AND.RH.GT.1.D0) RH = 1.D0                           DGRS4080
      GO TO 250                                                          DGRS4090
  235 NEWQ = L                                                           DGRS4100
      RH = 1.D0/PR3                                                      DGRS4110
      IF (RH.LT.1.1D0) GO TO 245                                         DGRS4120
      DO 240 I=1,N                                                       DGRS4130
  240 Y(I,NEWQ+1) = ERROR(I)*EL(L)/L                                     DGRS4140
      GO TO 255                                                          DGRS4150
  245 IDOUB = 10                                                         DGRS4160
      GO TO 290                                                          DGRS4170
  250 IF ((KFLAG.EQ.0).AND.(RH.LT.1.1D0)) GO TO 245                      DGRS4180
C                                                                        DGRS4190
C                                         IF THERE IS A CHANGE OF ORDER, RESET DGRS4200
C                                           NQ, L, AND THE COEFFICIENTS. IN    DGRS4210
C                                           ANY CASE H IS RESET ACCORDING TO   DGRS4220
C                                           RH AND THE Y ARRAY IS RESCALED.    DGRS4230
C                                           THEN EXIT FROM 285 IF THE STEP WAS DGRS4240
C                                           OK, OR REDO THE STEP OTHERWISE.    DGRS4250
      IF (NEWQ.EQ.NQ) GO TO 35                                           DGRS4260
  255 NQ = NEWQ                                                          DGRS4270
      L = NQ+1                                                           DGRS4280
      IRET = 2                                                           DGRS4290
      GO TO 15                                                           DGRS4300
C                                         CONTROL REACHES THIS SECTION IF 3 OR DGRS4310
C                                           MORE FAILURES HAVE OCCURED. IT IS  DGRS4320
```

```
C                                          ASSUMED THAT THE DERIVATIVES THAT   DGRS4330
C                                          HAVE ACCUMULATED IN THE Y ARRAY     DGRS4340
C                                          HAVE ERRORS OF THE WRONG ORDER.     DGRS4350
C                                          HENCE THE FIRST DERIVATIVE IS       DGRS4360
C                                          RECOMPUTED, AND THE ORDER IS SET    DGRS4370
C                                          TO 1. THEN H IS REDUCED BY A        DGRS4380
C                                          FACTOR OF 10, AND THE STEP IS       DGRS4390
C                                          RETRIED. AFTER A TOTAL OF 7         DGRS4400
C                                          FAILURES, AN EXIT IS TAKEN WITH     DGRS4410
C                                          KFLAG = -2.                         DGRS4420
  260 IF (KFLAG.EQ.-7) GO TO 275                                               DGRS4430
      RH = .1D0                                                                DGRS4440
      RH = DMAX1(HMIN/DABS(H),RH)                                              DGRS4450
      H = H*RH                                                                 DGRS4460
      CALL FCN (N,T,Y,SAVE1,eprime,eprime2)                                       +
      NFE = NFE+1                                                              DGRS4480
      DO 265 I=1,N                                                             DGRS4490
  265 Y(I,2) = H*SAVE1(I)                                                      DGRS4500
      IWEVAL = MITER                                                           DGRS4510
      IDOUB = 10                                                               DGRS4520
      IF (NQ.EQ.1) GO TO 50                                                    DGRS4530
      NQ = 1                                                                   DGRS4540
      L = 2                                                                    DGRS4550
      IRET = 3                                                                 DGRS4560
      GO TO 15                                                                 DGRS4570
C                                        ALL RETURNS ARE MADE THROUGH THIS     DGRS4580
C                                          SECTION. H IS SAVED IN HOLD TO      DGRS4590
C                                          ALLOW THE CALLER TO CHANGE H ON     DGRS4600
C                                          THE NEXT STEP.                      DGRS4610
  270 KFLAG = -1                                                               DGRS4620
      GO TO 290                                                                DGRS4630
  275 KFLAG = -2                                                               DGRS4640
      GO TO 290                                                                DGRS4650
  280 KFLAG = -3                                                               DGRS4660
      GO TO 290                                                                DGRS4670
  285 RMAX = 10.D0                                                             DGRS4680
  290 HOLD = H                                                                 DGRS4690
      JSTART = NQ                                                              DGRS4700
C--Diagnostic Check of first and second derivatives of E                          +
      if(tcum.eq.told)go to 310                                                   +
      write(8,300)tcum,step,y(1,1),y(2,1),y(3,1),y(4,1),y(5,1)                    +
  300 format(1x,e11.4,1x,I5,5(1x,e11.4))                                          +
      write(9,305)step,eprime,eprime2                                             +
  305 format(1x,I5,2(1x,e20.13))                                                  +
      RETURN                                                                   DGRS4710
      END                                                                      DGRS4720
```

```
C   IMSL ROUTINE NAME   - DGRCS
C
C-----------------------------------------------------------------------
C
C   COMPUTER            - IBM/DOUBLE
C
C   LATEST REVISION     - JANUARY 1, 1978
C
C   PURPOSE             - NUCLEUS CALLED ONLY BY IMSL SUBROUTINE DGEAR
C
C   PRECISION/HARDWARE  - SINGLE AND DOUBLE/H32
C                       - SINGLE/H36,H48,H60
C
C   REQD. IMSL ROUTINES - NONE REQUIRED
C
C   NOTATION            - INFORMATION ON SPECIAL NOTATION AND
C                           CONVENTIONS IS AVAILABLE IN THE MANUAL
C                           INTRODUCTION OR THROUGH IMSL ROUTINE UHELP
C
C   COPYRIGHT           - 1978 BY IMSL, INC. ALL RIGHTS RESERVED.
C
C   WARRANTY            - IMSL WARRANTS ONLY THAT IMSL TESTING HAS BEEN
C                           APPLIED TO THIS CODE. NO OTHER WARRANTY,
C                           EXPRESSED OR IMPLIED, IS APPLICABLE.
C
C-----------------------------------------------------------------------
C
      SUBROUTINE DGRCS  (METH,NQ,EL,TQ,MAXDER)
C                                  SPECIFICATIONS FOR ARGUMENTS
      INTEGER            METH,NQ,MAXDER
      REAL               TQ(1)
      DOUBLE PRECISION   EL(1)
C                                  SPECIFICATIONS FOR LOCAL VARIABLES
      INTEGER            K
      REAL               PERTST(12,2,3)
      DATA               PERTST/1.,1.,2.,1.,.3158,.7407E-1,
     1                   .1391E-1,.2182E-2,.2945E-3,.3492E-4,
     2                   .3692E-5,.3524E-6,1.,1.,.5,.1667,
     3                   .4167E-1,7*1.,2.,12.,24.,37.89,
     4                   53.33,70.08,87.97,106.9,126.7,
     5                   147.4,168.8,191.0,2.0,4.5,7.333,
     6                   10.42,13.7,7*1.,12.0,24.0,37.89,
     7                   53.33,70.08,87.97,106.9,126.7,
     8                   147.4,168.8,191.0,1.,3.0,6.0,
     9                   9.167,12.5,8*1./
C                                  FIRST EXECUTABLE STATEMENT
      GO TO (5,10), METH
    5 MAXDER = 12
      GO TO (15,20,25,30,35,40,45,50,55,60,65,70), NQ
   10 MAXDER = 5
      GO TO (75,80,85,90,95), NQ
C                                  THE FOLLOWING COEFFICIENTS SHOULD BE
C                                    DEFINED TO MACHINE ACCURACY. FOR A
C                                    GIVEN ORDER NQ, THEY CAN BE
C                                    CALCULATED BY USE OF THE
C                                    GENERATING POLYNOMIAL L(T), WHOSE
C                                    COEFFICIENTS ARE EL(I).. L(T) =
C                                    EL(1) + EL(2)*T + ... +
C                                    EL(NQ+1)*T**NQ. FOR THE IMPLICIT
C                                    ADAMS METHODS, L(T) IS GIVEN BY
C                                    DL/DT = (T+1)*(T+2)* ...
C                                    *(T+NQ-1)/K, L(-1) = 0, WHERE K =
```

```
C                                      FACTORIAL(NQ-1). FOR THE GEAR
C                                      METHODS, L(T) = (T+1)*(T+2)* ...
C                                      *(T+NQ)/K, WHERE K =
C                                      FACTORIAL(NQ)*(1 + 1/2 + ... +
C                                      1/NQ). THE ORDER IN WHICH THE
C                                      GROUPS APPEAR BELOW IS.. IMPLICIT
C                                      ADAMS METHODS OF ORDERS 1 TO 12,
C                                      BACKWARD DIFFERENTIATION METHODS
C                                      OF ORDERS 1 TO 5.
   15 EL(1) = 1.0D0
      GO TO 100
   20 EL(1) = 0.5D0
      EL(3) = 0.5D0
      GO TO 100
   25 EL(1) = 4.166666666666667D-01
      EL(3) = 0.75D0
      EL(4) = 1.666666666666667D-01
      GO TO 100
   30 EL(1) = 0.375D0
      EL(3) = 9.166666666666667D-01
      EL(4) = 3.333333333333333D-01
      EL(5) = 4.166666666666667D-02
      GO TO 100
   35 EL(1) = 3.486111111111111D-01
      EL(3) = 1.041666666666667D0
      EL(4) = 4.861111111111111D-01
      EL(5) = 1.041666666666667D-01
      EL(6) = 8.333333333333333D-03
      GO TO 100
   40 EL(1) = 3.298611111111111D-01
      EL(3) = 1.141666666666667D+00
      EL(4) = 0.625D+00
      EL(5) = 1.770833333333333D-01
      EL(6) = 0.025D+00
      EL(7) = 1.388888888888889D-03
      GO TO 100
   45 EL(1) = 3.155919312169312D-01
      EL(3) = 1.225D+00
      EL(4) = 7.518518518518519D-01
      EL(5) = 2.552083333333333D-01
      EL(6) = 4.861111111111111D-02
      EL(7) = 4.861111111111111D-03
      EL(8) = 1.984126984126984D-04
      GO TO 100
   50 EL(1) = 3.042245370370370D-01
      EL(3) = 1.296428571428571D+00
      EL(4) = 8.685185185185185D-01
      EL(5) = 3.357638888888889D-01
      EL(6) = 7.777777777777778D-02
      EL(7) = 1.064814814814815D-02
      EL(8) = 7.936507936507937D-04
      EL(9) = 2.480158730158730D-05
      GO TO 100
   55 EL(1) = 2.948680004409171D-01
      EL(3) = 1.358928571428571D+00
      EL(4) = 9.765542328042328D-01
      EL(5) = 4.171875D-01
      EL(6) = 1.113541666666667D-01
      EL(7) = 0.01875D+00
      EL(8) = 1.934523809523810D-03
      EL(9) = 1.116071428571429D-04
      EL(10)= 2.755731922398589D-06
```

```
      GO TO 100
   60 EL(1) = 2.869754464285714D-01
      EL(3) = 1.414484126984127D+00
      EL(4) = 1.077215608465609D+00
      EL(5) = 4.985670194003527D-01
      EL(6) = 1.484375D-01
      EL(7) = 2.906057098765432D-02
      EL(8) = 3.720238095238095D-03
      EL(9) = 2.996858465608466D-04
      EL(10)= 1.377865961199295D-05
      EL(11)= 2.755731922398589D-07
      GO TO 100
   65 EL(1) = 2.801895964439367D-01
      EL(3) = 1.464484126984127D+00
      EL(4) = 1.171514550264550D+00
      EL(5) = 5.793581900352734D-01
      EL(6) = 1.883228615520282D-01
      EL(7) = 4.143036265432099D-02
      EL(8) = 6.211144179894180D-03
      EL(9) = 6.252066798941799D-04
      EL(10)= 4.041740152851264D-05
      EL(11)= 1.515652557319224D-06
      EL(12)= 2.505210838544172D-08
      GO TO 100
   70 EL(1) = 2.742655400315991D-01
      EL(3) = 1.509938672438672D+00
      EL(4) = 1.260271164021164D+00
      EL(5) = 6.592341820987654D-01
      EL(6) = 2.304580026455027D-01
      EL(7) = 5.569724610523222D-02
      EL(8) = 9.439484126984127D-03
      EL(9) = 1.119274966931217D-03
      EL(10) = 9.093915343915344D-05
      EL(11) = 4.822530864197531D-06
      EL(12)= 1.503126503126503D-07
      EL(13)= 2.087675698786810D-09
      GO TO 100
C
   75 EL(1) = 1.0D+00
      GO TO 100
   80 EL(1) = 6.666666666666667D-01
      EL(3) = 3.333333333333333D-01
      GO TO 100
   85 EL(1) = 5.454545454545455D-01
      EL(3) = EL(1)
      EL(4) = 9.090909090909091D-02
      GO TO 100
   90 EL(1) = 0.48D+00
      EL(3) = 0.7D+00
      EL(4) = 0.2D+00
      EL(5) = 0.02D+00
      GO TO 100
   95 EL(1) = 4.379562043795620D-01
      EL(3) = 8.211678832116788D-01
      EL(4) = 3.102189781021898D-01
      EL(5) = 5.474452554744526D-02
      EL(6) = 3.649635036496350D-03
C
  100 DO 105 K=1,3
        TQ(K) = PERTST(NQ,METH,K)
  105 CONTINUE
      TQ(4) = .5D0*TQ(2)/(NQ+2)
```

```
      RETURN                                                    DGRC1870
      END                                                       DGRC1880
```

```
C   IMSL ROUTINE NAME   - DGRPS
C
C-----------------------------------------------------------------------
C
C   COMPUTER            - IBM/DOUBLE
C
C   LATEST REVISION     - NOVEMBER 1, 1984
C
C   PURPOSE             - NUCLEUS CALLED ONLY BY IMSL SUBROUTINE DGEAR
C
C   PRECISION/HARDWARE  - SINGLE AND DOUBLE/H32
C                       - SINGLE/H36,H48,H60
C
C   REQD. IMSL ROUTINES - LUDATF,LEQT1B,UERTST,UGETIO
C
C   NOTATION            - INFORMATION ON SPECIAL NOTATION AND
C                           CONVENTIONS IS AVAILABLE IN THE MANUAL
C                           INTRODUCTION OR THROUGH IMSL ROUTINE UHELP
C
C   COPYRIGHT           - 1984 BY IMSL, INC.  ALL RIGHTS RESERVED.
C
C   WARRANTY            - IMSL WARRANTS ONLY THAT IMSL TESTING HAS BEEN
C                           APPLIED TO THIS CODE.  NO OTHER WARRANTY,
C                           EXPRESSED OR IMPLIED, IS APPLICABLE.
C
C-----------------------------------------------------------------------
C
      SUBROUTINE DGRPS (FCN,FCNJ,Y,N0,CON,MITER,YMAX,SAVE1,SAVE2,PW,
     *                  EQUIL,IPIV,IER)
C                                  SPECIFICATIONS FOR ARGUMENTS
      INTEGER            N0,MITER,IPIV(1),IER
      DOUBLE PRECISION   Y(N0,1),CON,YMAX(1),SAVE1(1),SAVE2(1),PW(1),
     *                   EQUIL(1)
C                                  SPECIFICATIONS FOR LOCAL VARIABLES
C
      INTEGER           NC,MFC,KFLAG,JSTART,NQUSED,NSTEP,NFE,NJE,NPW,
     *                  NSQ,I,J1,J,NERROR,NSAVE1,NSAVE2,NEQUIL,NY,
     *                  IDUMMY(23),NLIM,II,IJ,LIM1,LIM2,NB,NLC,NUC,NWK
      REAL                SDUMMY(4)
      DOUBLE PRECISION  T,H,HMIN,HMAX,EPSC,UROUND,EPSJ,HUSED,D,R0,YJ,R,
     *                  D1,D2,WA,DUMMY(40)
      COMMON /DBAND/     NLC,NUC
      COMMON /GEAR/     T,H,HMIN,HMAX,EPSC,UROUND,EPSJ,HUSED,DUMMY,
     *                  SDUMMY,NC,MFC,KFLAG,JSTART,NSQ,NQUSED,NSTEP,
     *                  NFE,NJE,NPW,NERROR,NSAVE1,NSAVE2,NEQUIL,NY,
     *                   IDUMMY
C                                  THIS ROUTINE IS CALLED BY DGRST TO
C                                    COMPUTE AND PROCESS THE MATRIX P =
C                                    I - H*EL(1)*J , WHERE J IS AN
C                                    APPROXIMATION TO THE JACOBIAN. J
C                                    IS COMPUTED, EITHER BY THE USER-
C                                    SUPPLIED ROUTINE FCNJ IF MITER =
C                                    1, OR BY FINITE DIFFERENCING IF
C                                    MITER = 2. J IS STORED IN PW AND
C                                    REPLACED BY P, USING CON =
C                                    -H*EL(1). THEN P IS SUBJECTED TO
C                                    LU DECOMPOSITION IN PREPARATION
C                                    FOR LATER SOLUTION OF LINEAR
C                                    SYSTEMS WITH P AS COEFFICIENT
C                                    MATRIX. IN ADDITION TO VARIABLES
C                                    DESCRIBED PREVIOUSLY,
C                                    COMMUNICATION WITH DGRPS USES THE
```

```
C                                              FOLLOWING EPSJ = DSQRT(UROUND),
C                                              USED IN THE NUMERICAL JACOBIAN
C                                               INCREMENTS.
C
C                                            FIRST EXECUTABLE STATEMENT
      IF (NLC.EQ.-1) GO TO 45
C                                            BANDED JACOBIAN CASE
      NB = NLC+NUC+1
      NWK = NB*N0+1
      IF (MITER.EQ.2) GO TO 15
C                                             MITER = 1
      NLIM = NB*N0
      DO 5 I=1,NLIM
         PW(I) = 0.0D0
    5 CONTINUE
      CALL FCNJ(NC,T,Y,PW)
      DO 10 I=1,NLIM
         PW(I) = PW(I)*CON
   10 CONTINUE
      GO TO 35
C                                             MITER = 2
   15 D = 0.0D0
      DO 20 I=1,NC
   20 D = D+SAVE2(I)**2
      R0 = DABS(H)*DSQRT(D)*1.0D+03*UROUND
      DO 30 J=1,NC
         YJ = Y(J,1)
         R = EPSJ*YMAX(J)
         R = DMAX1(R,R0)
         Y(J,1) = Y(J,1)+R
         D = CON/R
         CALL FCN(NC,T,Y,SAVE1)
         LIM1 = MAX0(1,J-NUC)
         LIM2 = MIN0(N0,J+NLC)
         DO 25 I=LIM1,LIM2
            IJ = (J-I+NLC)*N0+I
            PW(IJ) = (SAVE1(I)-SAVE2(I))*D
   25    CONTINUE
         Y(J,1) = YJ
   30 CONTINUE
C                                             ADD IDENTITY MATRIX.
   35 DO 40 I=1,NC
         II = NLC*N0+I
         PW(II) = PW(II)+1.0D0
   40 CONTINUE
C                                             DO LU DECOMPOSITION ON P
C
      CALL LEQT1B(PW,NC,NLC,NUC,N0,EQUIL,1,N0,1,PW(NWK),IER)
      RETURN
C                                             FULL JACOBIAN CASE
   45 IF (MITER.EQ.2) GO TO 55
C                                             MITER = 1
      CALL FCNJ(NC,T,Y,PW)
      DO 50 I=1,NSQ
   50 PW(I) = PW(I)*CON
      GO TO 75
C                                             MITER = 2
   55 D = 0.0D0
      DO 60 I=1,NC
   60 D = D+SAVE2(I)**2
      R0 = DABS(H)*DSQRT(D)*1.0D+03*UROUND
      J1 = 0
```

```
      DO 70 J=1,NC                                                      DGRP1250
         YJ = Y(J,1)                                                    DGRP1260
         R = EPSJ*YMAX(J)                                               DGRP1270
         R = DMAX1(R,R0)                                                DGRP1280
         Y(J,1) = Y(J,1)+R                                              DGRP1290
         D = CON/R                                                      DGRP1300
         CALL FCN(NC,T,Y,SAVE1)                                         DGRP1310
         DO 65 I=1,NC                                                   DGRP1320
   65    PW(I+J1) = (SAVE1(I)-SAVE2(I))*D                               DGRP1330
         Y(J,1) = YJ                                                    DGRP1340
         J1 = J1+N0                                                     DGRP1350
   70 CONTINUE                                                          DGRP1360
C                                     ADD IDENTITY MATRIX.              DGRP1370
   75 J = 1                                                             DGRP1380
      DO 80 I=1,NC                                                      DGRP1390
         PW(J) = PW(J)+1.0D0                                            DGRP1400
         J = J+(N0+1)                                                   DGRP1410
   80 CONTINUE                                                          DGRP1420
C                                     DO LU DECOMPOSITION ON P.         DGRP1430
C                                                                       DGRP1440
      CALL LUDATF(PW,PW,NC,N0,0,D1,D2,IPIV,EQUIL,WA,IER)                DGRP1450
      RETURN                                                            DGRP1460
      END                                                               DGRP1470
```

```
C   IMSL ROUTINE NAME   - DGRIN
C
C-----------------------------------------------------------------------
C
C   COMPUTER            - IBM/DOUBLE
C
C   LATEST REVISION     - JANUARY 1, 1978
C
C   PURPOSE             - NUCLEUS CALLED ONLY BY IMSL SUBROUTINE DGEAR
C
C   PRECISION/HARDWARE  - SINGLE AND DOUBLE/H32
C                       - SINGLE/H36,H48,H60
C
C   REQD. IMSL ROUTINES - NONE REQUIRED
C
C   NOTATION            - INFORMATION ON SPECIAL NOTATION AND
C                           CONVENTIONS IS AVAILABLE IN THE MANUAL
C                           INTRODUCTION OR THROUGH IMSL ROUTINE UHELP
C
C   COPYRIGHT           - 1978 BY IMSL, INC. ALL RIGHTS RESERVED.
C
C   WARRANTY            - IMSL WARRANTS ONLY THAT IMSL TESTING HAS BEEN
C                           APPLIED TO THIS CODE. NO OTHER WARRANTY,
C                           EXPRESSED OR IMPLIED, IS APPLICABLE.
C
C-----------------------------------------------------------------------
C
      SUBROUTINE DGRIN  (TOUT,Y,N0,Y0)
C                                  SPECIFICATIONS FOR ARGUMENTS
      INTEGER            N0
      DOUBLE PRECISION   TOUT,Y0(N0),Y(N0,1)
C                                  SPECIFICATIONS FOR LOCAL VARIABLES
      INTEGER           NC,MFC,KFLAG,I,L,J,JSTART,NSQ,NQUSED,NSTEP,
     1                  NFE,NJE,NPW,NERROR,NSAVE1,NSAVE2,NEQUIL,NY,
     2                   IDUMMY(23)
      REAL               SDUMMY(4)
      DOUBLE PRECISION  T,H,HMIN,HMAX,EPSC,UROUND,EPSJ,HUSED,S,S1,
     1                   DUMMY(40)
      COMMON /GEAR/     T,H,HMIN,HMAX,EPSC,UROUND,EPSJ,HUSED,DUMMY,
     1                  SDUMMY,NC,MFC,KFLAG,JSTART,NSQ,NQUSED,NSTEP,
     2                  NFE,NJE,NPW,NERROR,NSAVE1,NSAVE2,NEQUIL,NY,
     3                   IDUMMY
C                                  FIRST EXECUTABLE STATEMENT         .
      DO 5 I = 1,NC
         Y0(I) = Y(I,1)
    5 CONTINUE
C                                  THIS SUBROUTINE COMPUTES INTERPOLATED
C                                    VALUES OF THE DEPENDENT VARIABLE
C                                    Y AND STORES THEM IN Y0. THE
C                                    INTERPOLATION IS TO THE
C                                    POINT T = TOUT, AND USES THE
C                                    NORDSIECK HISTORY ARRAY Y, AS
C                                    FOLLOWS..
C                                                NQ
C                                    Y0(I)  =   SUM  Y(I,J+1)*S**J ,
C                                               J=0
C                                    WHERE S = -(T-TOUT)/H.
      L = JSTART + 1
      S = (TOUT - T)/H
      S1 = 1.0D0
      DO 15 J = 2,L
         S1 = S1*S
```

```
      DO 10 I = 1,NC
         Y0(I) = Y0(I) + S1*Y(I,J)
10    CONTINUE
15 CONTINUE
   RETURN
   END
```

```
C   IMSL ROUTINE NAME   - LUDATF                                       LUDA0010
C                                                                       LUDA0020
C-----------------------------------------------------------------------LUDA0030
C                                                                       LUDA0040
C   COMPUTER            - IBM/DOUBLE                                    LUDA0050
C                                                                       LUDA0060
C   LATEST REVISION     - JANUARY 1, 1978                               LUDA0070
C                                                                       LUDA0080
C   PURPOSE             - L-U DECOMPOSITION BY THE CROUT ALGORITHM      LUDA0090
C                           WITH OPTIONAL ACCURACY TEST.                LUDA0100
C                                                                       LUDA0110
C   USAGE               - CALL LUDATF (A,LU,N,IA,IDGT,D1,D2,IPVT,       LUDA0120
C                           EQUIL,WA,IER)                               LUDA0130
C                                                                       LUDA0140
C   ARGUMENTS    A      - INPUT MATRIX OF DIMENSION N BY N CONTAINING   LUDA0150
C                           THE MATRIX TO BE DECOMPOSED.                LUDA0160
C                LU     - REAL OUTPUT MATRIX OF DIMENSION N BY N        LUDA0170
C                           CONTAINING THE L-U DECOMPOSITION OF A       LUDA0180
C                           ROWWISE PERMUTATION OF THE INPUT MATRIX.    LUDA0190
C                           FOR A DESCRIPTION OF THE FORMAT OF LU, SEE  LUDA0200
C                           EXAMPLE.                                    LUDA0210
C                N      - INPUT SCALAR CONTAINING THE ORDER OF THE      LUDA0220
C                           MATRIX A.                                   LUDA0230
C                IA     - INPUT SCALAR CONTAINING THE ROW DIMENSION OF  LUDA0240
C                           MATRICES A AND LU EXACTLY AS SPECIFIED IN   LUDA0250
C                           THE CALLING PROGRAM.                        LUDA0260
C                IDGT   - INPUT OPTION.                                 LUDA0270
C                           IF IDGT IS GREATER THAN ZERO, THE NON-ZERO  LUDA0280
C                           ELEMENTS OF A ARE ASSUMED TO BE CORRECT TO  LUDA0290
C                           IDGT DECIMAL PLACES.  LUDATF PERFORMS AN    LUDA0300
C                           ACCURACY TEST TO DETERMINE IF THE COMPUTED  LUDA0310
C                           DECOMPOSITION IS THE EXACT DECOMPOSITION    LUDA0320
C                           OF A MATRIX WHICH DIFFERS FROM THE GIVEN    LUDA0330
C                           ONE BY LESS THAN ITS UNCERTAINTY.           LUDA0340
C                         IF IDGT IS EQUAL TO ZERO, THE ACCURACY TEST   LUDA0350
C                           IS BYPASSED.                                LUDA0360
C                D1     - OUTPUT SCALAR CONTAINING ONE OF THE TWO       LUDA0370
C                           COMPONENTS OF THE DETERMINANT. SEE          LUDA0380
C                           DESCRIPTION OF PARAMETER D2, BELOW.         LUDA0390
C                D2     - OUTPUT SCALAR CONTAINING ONE OF THE           LUDA0400
C                           TWO COMPONENTS OF THE DETERMINANT. THE      LUDA0410
C                           DETERMINANT MAY BE EVALUATED AS (D1)(2**D2).LUDA0420
C                IPVT   - OUTPUT VECTOR OF LENGTH N CONTAINING THE      LUDA0430
C                           PERMUTATION INDICES. SEE DOCUMENT           LUDA0440
C                           (ALGORITHM).                                LUDA0450
C                EQUIL  - OUTPUT VECTOR OF LENGTH N CONTAINING          LUDA0460
C                           RECIPROCALS OF THE ABSOLUTE VALUES OF       LUDA0470
C                           THE LARGEST (IN ABSOLUTE VALUE) ELEMENT     LUDA0480
C                           IN EACH ROW.                                LUDA0490
C                WA     - ACCURACY TEST PARAMETER, OUTPUT ONLY IF       LUDA0500
C                           IDGT IS GREATER THAN ZERO.                  LUDA0510
C                           SEE ELEMENT DOCUMENTATION FOR DETAILS.      LUDA0520
C                IER    - ERROR PARAMETER. (OUTPUT)                     LUDA0530
C                         TERMINAL ERROR                                LUDA0540
C                           IER = 129 INDICATES THAT MATRIX A IS        LUDA0550
C                             ALGORITHMICALLY SINGULAR. (SEE THE        LUDA0560
C                             CHAPTER L PRELUDE).                       LUDA0570
C                         WARNING ERROR                                 LUDA0580
C                           IER = 34 INDICATES THAT THE ACCURACY TEST   LUDA0590
C                             FAILED.  THE COMPUTED SOLUTION MAY BE IN  LUDA0600
C                             ERROR BY MORE THAN CAN BE ACCOUNTED FOR   LUDA0610
C                             BY THE UNCERTAINTY OF THE DATA.  THIS     LUDA0620
```

```
C                            WARNING CAN BE PRODUCED ONLY IF IDGT IS
C                            GREATER THAN 0 ON INPUT.  SEE CHAPTER L
C                            PRELUDE FOR FURTHER DISCUSSION.
C
C   PRECISION/HARDWARE  - SINGLE AND DOUBLE/H32
C                       - SINGLE/H36,H48,H60
C
C   REQD. IMSL ROUTINES - UERTST,UGETIO
C
C   NOTATION            - INFORMATION ON SPECIAL NOTATION AND
C                           CONVENTIONS IS AVAILABLE IN THE MANUAL
C                           INTRODUCTION OR THROUGH IMSL ROUTINE UHELP
C
C   REMARKS      A TEST FOR SINGULARITY IS MADE AT TWO LEVELS:
C                1.  A ROW OF THE ORIGINAL MATRIX A IS NULL.
C                2.  A COLUMN BECOMES NULL IN THE FACTORIZATION PROCESS.
C
C   COPYRIGHT           - 1978 BY IMSL, INC. ALL RIGHTS RESERVED.
C
C   WARRANTY            - IMSL WARRANTS ONLY THAT IMSL TESTING HAS BEEN
C                           APPLIED TO THIS CODE. NO OTHER WARRANTY,
C                           EXPRESSED OR IMPLIED, IS APPLICABLE.
C
C-----------------------------------------------------------------------
C
      SUBROUTINE LUDATF (A,LU,N,IA,IDGT,D1,D2,IPVT,EQUIL,WA,IER)
C
      DIMENSION          A(IA,1),LU(IA,1),IPVT(1),EQUIL(1)
      DOUBLE PRECISION   A,LU,D1,D2,EQUIL,WA,ZERO,ONE,FOUR,SIXTN,SIXTH,
     *                   RN,WREL,BIGA,BIG,P,SUM,AI,WI,T,TEST,Q
      DATA               ZERO,ONE,FOUR,SIXTN,SIXTH/0.D0,1.D0,4.D0,
     *                    16.D0,.0625D0/
C                                  FIRST EXECUTABLE STATEMENT
C                                  INITIALIZATION
      IER = 0
      RN = N
      WREL = ZERO
      D1 = ONE
      D2 = ZERO
      BIGA = ZERO
      DO 10 I=1,N
         BIG = ZERO
         DO 5 J=1,N
            P = A(I,J)
            LU(I,J) = P
            P = DABS(P)
            IF (P .GT. BIG) BIG = P
    5    CONTINUE
         IF (BIG .GT. BIGA) BIGA = BIG
         IF (BIG .EQ. ZERO) GO TO 110
         EQUIL(I) = ONE/BIG
   10 CONTINUE
      DO 105 J=1,N
         JM1 = J-1
         IF (JM1 .LT. 1) GO TO 40
C                                  COMPUTE U(I,J), I=1,...,J-1
         DO 35 I=1,JM1
            SUM = LU(I,J)
            IM1 = I-1
            IF (IDGT .EQ. 0) GO TO 25
C                                  WITH ACCURACY TEST
            AI = DABS(SUM)
```

```
            WI = ZERO                                                   LUDA1250
            IF (IM1 .LT. 1) GO TO 20                                    LUDA1260
            DO 15 K=1,IM1                                               LUDA1270
               T = LU(I,K)*LU(K,J)                                      LUDA1280
               SUM = SUM-T                                              LUDA1290
               WI = WI+DABS(T)                                          LUDA1300
   15       CONTINUE                                                    LUDA1310
            LU(I,J) = SUM                                               LUDA1320
   20       WI = WI+DABS(SUM)                                           LUDA1330
            IF (AI .EQ. ZERO) AI = BIGA                                 LUDA1340
            TEST = WI/AI                                                LUDA1350
            IF (TEST .GT. WREL) WREL = TEST                             LUDA1360
            GO TO 35                                                    LUDA1370
C                                  WITHOUT ACCURACY                     LUDA1380
   25       IF (IM1 .LT. 1) GO TO 35                                    LUDA1390
            DO 30 K=1,IM1                                               LUDA1400
               SUM = SUM-LU(I,K)*LU(K,J)                                LUDA1410
   30       CONTINUE                                                    LUDA1420
            LU(I,J) = SUM                                               LUDA1430
   35    CONTINUE                                                       LUDA1440
   40    P = ZERO                                                       LUDA1450
C                                  COMPUTE U(J,J) AND L(I,J), I=J+1,...,LUDA1460
         DO 70 I=J,N                                                    LUDA1470
            SUM = LU(I,J)                                               LUDA1480
            IF (IDGT .EQ. 0) GO TO 55                                   LUDA1490
C                                  WITH ACCURACY TEST                   LUDA1500
            AI = DABS(SUM)                                              LUDA1510
            WI = ZERO                                                   LUDA1520
            IF (JM1 .LT. 1) GO TO 50                                    LUDA1530
            DO 45 K=1,JM1                                               LUDA1540
               T = LU(I,K)*LU(K,J)                                      LUDA1550
               SUM = SUM-T                                              LUDA1560
               WI = WI+DABS(T)                                          LUDA1570
   45       CONTINUE                                                    LUDA1580
            LU(I,J) = SUM                                               LUDA1590
   50       WI = WI+DABS(SUM)                                           LUDA1600
            IF (AI .EQ. ZERO) AI = BIGA                                 LUDA1610
            TEST = WI/AI                                                LUDA1620
            IF (TEST .GT. WREL) WREL = TEST                             LUDA1630
            GO TO 65                                                    LUDA1640
C                                  WITHOUT ACCURACY TEST                LUDA1650
   55       IF (JM1 .LT. 1) GO TO 65                                    LUDA1660
            DO 60 K=1,JM1                                               LUDA1670
               SUM = SUM-LU(I,K)*LU(K,J)                                LUDA1680
   60       CONTINUE                                                    LUDA1690
            LU(I,J) = SUM                                               LUDA1700
   65       Q = EQUIL(I)*DABS(SUM)                                      LUDA1710
            IF (P .GE. Q) GO TO 70                                      LUDA1720
            P = Q                                                       LUDA1730
            IMAX = I                                                    LUDA1740
   70    CONTINUE                                                       LUDA1750
C                                  TEST FOR ALGORITHMIC SINGULARITY     LUDA1760
         IF (RN+P .EQ. RN) GO TO 110                                    LUDA1770
         IF (J .EQ. IMAX) GO TO 80                                      LUDA1780
C                                  INTERCHANGE ROWS J AND IMAX          LUDA1790
         D1 = -D1                                                       LUDA1800
         DO 75 K=1,N                                                    LUDA1810
            P = LU(IMAX,K)                                              LUDA1820
            LU(IMAX,K) = LU(J,K)                                        LUDA1830
            LU(J,K) = P                                                 LUDA1840
   75    CONTINUE                                                       LUDA1850
         EQUIL(IMAX) = EQUIL(J)                                         LUDA1860
```

```
   80    IPVT(J) = IMAX
         D1 = D1*LU(J,J)
   85    IF (DABS(D1) .LE. ONE) GO TO 90
         D1 = D1*SIXTH
         D2 = D2+FOUR
         GO TO 85
   90    IF (DABS(D1) .GE. SIXTH) GO TO 95
         D1 = D1*SIXTN
         D2 = D2-FOUR
         GO TO 90
   95    CONTINUE
         JP1 = J+1
         IF (JP1 .GT. N) GO TO 105
C                                  DIVIDE BY PIVOT ELEMENT U(J,J)
         P = LU(J,J)
         DO 100 I=JP1,N
            LU(I,J) = LU(I,J)/P
  100    CONTINUE
  105 CONTINUE
C                                  PERFORM ACCURACY TEST
      IF (IDGT .EQ. 0) GO TO 9005
      P = 3*N+3
      WA = P*WREL
      IF (WA+10.D0**(-IDGT) .NE. WA) GO TO 9005
      IER = 34
      GO TO 9000
C                                  ALGORITHMIC SINGULARITY
  110 IER = 129
      D1 = ZERO
      D2 = ZERO
 9000 CONTINUE
C                                  PRINT ERROR
      CALL UERTST(IER,6HLUDATF)
 9005 RETURN
      END
```

```
C   IMSL ROUTINE NAME   - LUELMF
C
C-----------------------------------------------------------------------
C
C   COMPUTER            - IBM/DOUBLE
C
C   LATEST REVISION     - JANUARY 1, 1978
C
C   PURPOSE             - ELIMINATION PART OF SOLUTION OF AX=B
C                           (FULL STORAGE MODE)
C
C   USAGE               - CALL LUELMF (A,B,IPVT,N,IA,X)
C
C   ARGUMENTS    A      - A = LU (THE RESULT COMPUTED IN THE IMSL
C                           ROUTINE LUDATF) WHERE L IS A LOWER
C                           TRIANGULAR MATRIX WITH ONES ON THE MAIN
C                           DIAGONAL. U IS UPPER TRIANGULAR. L AND U
C                           ARE STORED AS A SINGLE MATRIX A AND THE
C                           UNIT DIAGONAL OF L IS NOT STORED. (INPUT)
C                B      - B IS A VECTOR OF LENGTH N ON THE RIGHT HAND
C                           SIDE OF THE EQUATION AX=B. (INPUT)
C                IPVT   - THE PERMUTATION MATRIX RETURNED FROM THE
C                           IMSL ROUTINE LUDATF, STORED AS AN N LENGTH
C                           VECTOR. (INPUT)
C                N      - ORDER OF A AND NUMBER OF ROWS IN B. (INPUT)
C                IA     - ROW DIMENSION OF A EXACTLY AS SPECIFIED IN
C                           THE DIMENSION STATEMENT IN THE CALLING
C                           PROGRAM. (INPUT)
C                X      - THE RESULT X. (OUTPUT)
C
C   PRECISION/HARDWARE  - SINGLE AND DOUBLE/H32
C                       - SINGLE/H36,H48,H60
C
C   REQD. IMSL ROUTINES - NONE REQUIRED
C
C   NOTATION            - INFORMATION ON SPECIAL NOTATION AND
C                           CONVENTIONS IS AVAILABLE IN THE MANUAL
C                           INTRODUCTION OR THROUGH IMSL ROUTINE UHELP
C
C   COPYRIGHT           - 1978 BY IMSL, INC. ALL RIGHTS RESERVED.
C
C   WARRANTY            - IMSL WARRANTS ONLY THAT IMSL TESTING HAS BEEN
C                           APPLIED TO THIS CODE. NO OTHER WARRANTY,
C                           EXPRESSED OR IMPLIED, IS APPLICABLE.
C
C-----------------------------------------------------------------------
C
      SUBROUTINE LUELMF (A,B,IPVT,N,IA,X)
C
      DIMENSION          A(IA,1),B(1),IPVT(1),X(1)
      DOUBLE PRECISION   A,B,X,SUM
C                                  FIRST EXECUTABLE STATEMENT
C                                  SOLVE LY = B FOR Y
      DO 5 I=1,N
    5 X(I) = B(I)
      IW = 0
      DO 20 I=1,N
         IP = IPVT(I)
         SUM = X(IP)
         X(IP) = X(I)
         IF (IW .EQ. 0) GO TO 15
         IM1 = I-1
```

```
         DO 10 J=IW,IM1                                                 LUEF0630
            SUM = SUM-A(I,J)*X(J)                                       LUEF0640
   10    CONTINUE                                                       LUEF0650
         GO TO 20                                                       LUEF0660
   15    IF (SUM .NE. 0.D0) IW = I                                      LUEF0670
   20 X(I) = SUM                                                        LUEF0680
C                             SOLVE UX = Y FOR X                        LUEF0690
      DO 30 IB=1,N                                                      LUEF0700
         I = N+1-IB                                                     LUEF0710
         IP1 = I+1                                                      LUEF0720
         SUM = X(I)                                                     LUEF0730
         IF (IP1 .GT. N) GO TO 30                                       LUEF0740
         DO 25 J=IP1,N                                                  LUEF0750
            SUM = SUM-A(I,J)*X(J)                                       LUEF0760
   25    CONTINUE                                                       LUEF0770
   30 X(I) = SUM/A(I,I)                                                 LUEF0780
      RETURN                                                            LUEF0790
      END                                                               LUEF0800
```

```
C   IMSL ROUTINE NAME   - LEQT1B
C
C-----------------------------------------------------------------------
C
C   COMPUTER            - IBM/DOUBLE
C
C   LATEST REVISION     - JANUARY 1, 1978
C
C   PURPOSE             - LINEAR EQUATION SOLUTION - BAND STORAGE
C                           MODE - SPACE ECONOMIZER SOLUTION
C
C   USAGE               - CALL LEQT1B (A,N,NLC,NUC,IA,B,M,IB,IJOB,XL,
C                           IER)
C
C   ARGUMENTS    A      - INPUT/OUTPUT MATRIX OF DIMENSION N BY
C                           (NUC+NLC+1). SEE PARAMETER IJOB.
C                N      - ORDER OF MATRIX A AND THE NUMBER OF ROWS IN
C                           B. (INPUT)
C                NLC    - NUMBER OF LOWER CODIAGONALS IN MATRIX A.
C                           (INPUT)
C                NUC    - NUMBER OF UPPER CODIAGONALS IN MATRIX A.
C                           (INPUT)
C                IA     - ROW DIMENSION OF MATRIX A EXACTLY AS
C                           SPECIFIED IN THE DIMENSION STATEMENT IN THE
C                           CALLING PROGRAM. (INPUT)
C                B      - INPUT/OUTPUT MATRIX OF DIMENSION N BY M.
C                           ON INPUT, B CONTAINS THE M RIGHT-HAND SIDES
C                           OF THE EQUATION AX = B. ON OUTPUT, THE
C                           SOLUTION MATRIX X REPLACES B. IF IJOB = 1,
C                           B IS NOT USED.
C                M      - NUMBER OF RIGHT HAND SIDES (COLUMNS IN B).
C                           (INPUT)
C                IB     - ROW DIMENSION OF MATRIX B EXACTLY AS
C                           SPECIFIED IN THE DIMENSION STATEMENT IN THE
C                           CALLING PROGRAM. (INPUT)
C                IJOB   - INPUT OPTION PARAMETER. IJOB = I IMPLIES WHEN
C                           I = 0, FACTOR THE MATRIX A AND SOLVE THE
C                             EQUATION AX = B. ON INPUT, A CONTAINS THE
C                             COEFFICIENT MATRIX OF THE EQUATION AX = B,
C                             WHERE A IS ASSUMED TO BE AN N BY N BAND
C                             MATRIX. A IS STORED IN BAND STORAGE MODE
C                             AND THEREFORE HAS DIMENSION N BY
C                             (NLC+NUC+1). ON OUTPUT, A IS REPLACED
C                             BY THE U MATRIX OF THE L-U DECOMPOSITION
C                             OF A ROWWISE PERMUTATION OF MATRIX A. U
C                             IS STORED IN BAND STORAGE MODE.
C                           I = 1, FACTOR THE MATRIX A. A CONTAINS THE
C                             SAME INPUT/OUTPUT INFORMATION AS IF
C                             IJOB = 0.
C                           I = 2, SOLVE THE EQUATION AX = B. THIS
C                             OPTION IMPLIES THAT LEQT1B HAS ALREADY
C                             BEEN CALLED USING IJOB = 0 OR 1 SO THAT
C                             THE MATRIX A HAS ALREADY BEEN FACTORED.
C                             IN THIS CASE, OUTPUT MATRICES A AND XL
C                             MUST HAVE BEEN SAVED FOR REUSE IN THE
C                             CALL TO LEQT1B.
C                XL     - WORK AREA OF DIMENSION N*(NLC+1). THE FIRST
C                           NLC*N LOCATIONS OF XL CONTAIN COMPONENTS OF
C                           THE L MATRIX OF THE L-U DECOMPOSITION OF A
C                           ROWWISE PERMUTATION OF A. THE LAST N
C                           LOCATIONS CONTAIN THE PIVOT INDICES.
C                IER    - ERROR PARAMETER. (OUTPUT)
```

```
C                               TERMINAL ERROR
C                                 IER = 129 INDICATES THAT MATRIX A IS
C                                   ALGORITHMICALLY SINGULAR. (SEE THE
C                                   CHAPTER L PRELUDE).
C
C   PRECISION/HARDWARE  - SINGLE AND DOUBLE/H32
C                       - SINGLE/H36,H48,H60
C
C   REQD. IMSL ROUTINES - UERTST,UGETIO
C
C   NOTATION            - INFORMATION ON SPECIAL NOTATION AND
C                           CONVENTIONS IS AVAILABLE IN THE MANUAL
C                           INTRODUCTION OR THROUGH IMSL ROUTINE UHELP
C
C   COPYRIGHT           - 1978 BY IMSL, INC. ALL RIGHTS RESERVED.
C
C   WARRANTY            - IMSL WARRANTS ONLY THAT IMSL TESTING HAS BEEN
C                           APPLIED TO THIS CODE. NO OTHER WARRANTY,
C                           EXPRESSED OR IMPLIED, IS APPLICABLE.
C
C-----------------------------------------------------------------------
C
      SUBROUTINE LEQT1B (A,N,NLC,NUC,IA,B,M,IB,IJOB,XL,IER)
C
      DIMENSION          A(IA,1),XL(N,1),B(IB,1)
      DOUBLE PRECISION   A,XL,B,P,Q,ZERO,ONE,RN
      DATA               ZERO/0.D0/,ONE/1.0D0/
C                                  FIRST EXECUTABLE STATEMENT
      IER = 0
      JBEG = NLC+1
      NLC1 = JBEG
      IF (IJOB .EQ. 2) GO TO 80
      RN = N
C                                  RESTRUCTURE THE MATRIX
C                                  FIND RECIPROCAL OF THE LARGEST
C                                  ABSOLUTE VALUE IN ROW I
      I = 1
      NC = JBEG+NUC
      NN = NC
      JEND = NC
      IF (N .EQ. 1 .OR. NLC .EQ. 0) GO TO 25
    5 K = 1
      P = ZERO
      DO 10 J = JBEG,JEND
         A(I,K) = A(I,J)
         Q =  DABS(A(I,K))
         IF (Q .GT. P) P = Q
         K = K+1
   10 CONTINUE
      IF (P .EQ. ZERO) GO TO 135
      XL(I,NLC1) = ONE/P
      IF (K .GT. NC) GO TO 20
      DO 15 J = K,NC
         A(I,J) = ZERO
   15 CONTINUE
   20 I = I+1
      JBEG = JBEG-1
      IF (JEND-JBEG .EQ. N) JEND = JEND-1
      IF (I .LE. NLC) GO TO 5
      JBEG = I
      NN = JEND
   25 JEND = N-NUC
```

```
      DO 40 I = JBEG,N                                                  LE1B1250
         P = ZERO                                                       LE1B1260
         DO 30 J = 1,NN                                                 LE1B1270
            Q =  DABS(A(I,J))                                           LE1B1280
            IF (Q .GT. P) P = Q                                         LE1B1290
   30    CONTINUE                                                       LE1B1300
         IF (P .EQ. ZERO) GO TO 135                                     LE1B1310
         XL(I,NLC1) = ONE/P                                             LE1B1320
         IF (I .EQ. JEND) GO TO 37                                      LE1B1330
         IF (I .LT. JEND) GO TO 40                                      LE1B1340
         K = NN+1                                                       LE1B1350
         DO 35 J = K,NC                                                 LE1B1360
            A(I,J) = ZERO                                               LE1B1370
   35    CONTINUE                                                       LE1B1380
   37    NN = NN-1                                                      LE1B1390
   40 CONTINUE                                                          LE1B1400
      L = NLC                                                           LE1B1410
C                                   L-U DECOMPOSITION                   LE1B1420
      DO 75 K = 1,N                                                     LE1B1430
         P =  DABS(A(K,1))*XL(K,NLC1)                                   LE1B1440
         I = K                                                          LE1B1450
         IF (L .LT. N) L = L+1                                          LE1B1460
         K1 = K+1                                                       LE1B1470
         IF (K1 .GT. L) GO TO 50                                        LE1B1480
         DO 45 J = K1,L                                                 LE1B1490
            Q = DABS(A(J,1))*XL(J,NLC1)                                 LE1B1500
            IF (Q .LE. P) GO TO 45                                      LE1B1510
            P = Q                                                       LE1B1520
            I = J                                                       LE1B1530
   45    CONTINUE                                                       LE1B1540
   50    XL(I,NLC1) = XL(K,NLC1)                                        LE1B1550
         XL(K,NLC1) = I                                                 LE1B1560
C                                   SINGULARITY FOUND                   LE1B1570
         Q = RN+P                                                       LE1B1580
         IF (Q .EQ. RN) GO TO 135                                       LE1B1590
C                                   INTERCHANGE ROWS I AND K            LE1B1600
         IF (K .EQ. I) GO TO 60                                         LE1B1610
         DO 55 J = 1,NC                                                 LE1B1620
            P = A(K,J)                                                  LE1B1630
            A(K,J) = A(I,J)                                             LE1B1640
            A(I,J) = P                                                  LE1B1650
   55    CONTINUE                                                       LE1B1660
   60    IF (K1 .GT. L) GO TO 75                                        LE1B1670
         DO 70 I = K1,L                                                 LE1B1680
            P = A(I,1)/A(K,1)                                           LE1B1690
            IK = I-K                                                    LE1B1700
            XL(K1,IK) = P                                               LE1B1710
            DO 65 J = 2,NC                                              LE1B1720
               A(I,J-1) = A(I,J)-P*A(K,J)                               LE1B1730
   65    CONTINUE                                                       LE1B1740
         A(I,NC) = ZERO                                                 LE1B1750
   70    CONTINUE                                                       LE1B1760
   75 CONTINUE                                                          LE1B1770
      IF (IJOB .EQ. 1) GO TO 9005                                       LE1B1780
C                                   FORWARD SUBSTITUTION                LE1B1790
   80 L = NLC                                                           LE1B1800
      DO 105 K = 1,N                                                    LE1B1810
         I = XL(K,NLC1)                                                 LE1B1820
         IF (I .EQ. K) GO TO 90                                         LE1B1830
         DO 85 J = 1,M                                                  LE1B1840
            P = B(K,J)                                                  LE1B1850
            B(K,J) = B(I,J)                                             LE1B1860
```

```
                B(I,J) = P
   85        CONTINUE
   90        IF (L .LT. N) L = L+1
             K1 = K+1
             IF (K1 .GT. L) GO TO 105
             DO 100 I = K1,L
                IK = I-K
                P = XL(K1,IK)
                DO 95 J = 1,M
                   B(I,J) = B(I,J)-P*B(K,J)
   95           CONTINUE
  100        CONTINUE
  105     CONTINUE
C                                  BACKWARD SUBSTITUTION
          JBEG = NUC+NLC
          DO 125 J = 1,M
             L = 1
             K1 = N+1
             DO 120 I = 1,N
                K = K1-I
                P = B(K,J)
                IF (L .EQ. 1) GO TO 115
                DO 110 KK = 2,L
                   IK = KK+K
                   P = P-A(K,KK)*B(IK-1,J)
  110           CONTINUE
  115           B(K,J) = P/A(K,1)
                IF (L .LE. JBEG) L = L+1
  120        CONTINUE
  125     CONTINUE
          GO TO 9005
  135     IER = 129
 9000     CONTINUE
          CALL UERTST(IER,6HLEQT1B)
 9005     RETURN
          END
```

```
C   IMSL ROUTINE NAME   - UERTST
C
C-----------------------------------------------------------------------
C
C   COMPUTER            - IBM/SINGLE
C
C   LATEST REVISION     - JUNE 1, 1982
C
C   PURPOSE             - PRINT A MESSAGE REFLECTING AN ERROR CONDITION
C
C   USAGE               - CALL UERTST (IER,NAME)
C
C   ARGUMENTS    IER    - ERROR PARAMETER. (INPUT)
C                           IER = I+J WHERE
C                             I = 128 IMPLIES TERMINAL ERROR MESSAGE,
C                             I =  64 IMPLIES WARNING WITH FIX MESSAGE,
C                             I =  32 IMPLIES WARNING MESSAGE.
C                             J = ERROR CODE RELEVANT TO CALLING
C                                 ROUTINE.
C                NAME   - A CHARACTER STRING OF LENGTH SIX PROVIDING
C                           THE NAME OF THE CALLING ROUTINE. (INPUT)
C
C   PRECISION/HARDWARE  - SINGLE/ALL
C
C   REQD. IMSL ROUTINES - UGETIO,USPKD
C
C   NOTATION            - INFORMATION ON SPECIAL NOTATION AND
C                           CONVENTIONS IS AVAILABLE IN THE MANUAL
C                           INTRODUCTION OR THROUGH IMSL ROUTINE UHELP
C
C   REMARKS      THE ERROR MESSAGE PRODUCED BY UERTST IS WRITTEN
C                TO THE STANDARD OUTPUT UNIT. THE OUTPUT UNIT
C                NUMBER CAN BE DETERMINED BY CALLING UGETIO AS
C                FOLLOWS..   CALL UGETIO(1,NIN,NOUT).
C                THE OUTPUT UNIT NUMBER CAN BE CHANGED BY CALLING
C                UGETIO AS FOLLOWS..
C                                NIN = 0
C                                NOUT = NEW OUTPUT UNIT NUMBER
C                                CALL UGETIO(3,NIN,NOUT)
C                SEE THE UGETIO DOCUMENT FOR MORE DETAILS.
C
C   COPYRIGHT           - 1982 BY IMSL, INC. ALL RIGHTS RESERVED.
C
C   WARRANTY            - IMSL WARRANTS ONLY THAT IMSL TESTING HAS BEEN
C                           APPLIED TO THIS CODE. NO OTHER WARRANTY,
C                           EXPRESSED OR IMPLIED, IS APPLICABLE.
C
C-----------------------------------------------------------------------
C
      SUBROUTINE UERTST (IER,NAME)
C                                  SPECIFICATIONS FOR ARGUMENTS
      INTEGER            IER
      INTEGER            NAME(1)
C                                  SPECIFICATIONS FOR LOCAL VARIABLES
      INTEGER            I,IEQ,IEQDF,IOUNIT,LEVEL,LEVOLD,NAMEQ(6),
     *                   NAMSET(6),NAMUPK(6),NIN,NMTB
      DATA               NAMSET/1HU,1HE,1HR,1HS,1HE,1HT/
      DATA               NAMEQ/6*1H /
      DATA               LEVEL/4/,IEQDF/0/,IEQ/1H=/
C                                  UNPACK NAME INTO NAMUPK
C                                  FIRST EXECUTABLE STATEMENT
      CALL USPKD (NAME,6,NAMUPK,NMTB)
```

```
C                                 GET OUTPUT UNIT NUMBER                UERT0630
      CALL UGETIO(1,NIN,IOUNIT)                                         UERT0640
C                                 CHECK IER                             UERT0650
      IF (IER.GT.999) GO TO 25                                          UERT0660
      IF (IER.LT.-32) GO TO 55                                          UERT0670
      IF (IER.LE.128) GO TO 5                                           UERT0680
      IF (LEVEL.LT.1) GO TO 30                                          UERT0690
C                                 PRINT TERMINAL MESSAGE                UERT0700
      IF (IEQDF.EQ.1) WRITE(IOUNIT,35) IER,NAMEQ,IEQ,NAMUPK             UERT0710
      IF (IEQDF.EQ.0) WRITE(IOUNIT,35) IER,NAMUPK                       UERT0720
      GO TO 30                                                          UERT0730
    5 IF (IER.LE.64) GO TO 10                                           UERT0740
      IF (LEVEL.LT.2) GO TO 30                                          UERT0750
C                                 PRINT WARNING WITH FIX MESSAGE        UERT0760
      IF (IEQDF.EQ.1) WRITE(IOUNIT,40) IER,NAMEQ,IEQ,NAMUPK             UERT0770
      IF (IEQDF.EQ.0) WRITE(IOUNIT,40) IER,NAMUPK                       UERT0780
      GO TO 30                                                          UERT0790
   10 IF (IER.LE.32) GO TO 15                                           UERT0800
C                                 PRINT WARNING MESSAGE                 UERT0810
      IF (LEVEL.LT.3) GO TO 30                                          UERT0820
      IF (IEQDF.EQ.1) WRITE(IOUNIT,45) IER,NAMEQ,IEQ,NAMUPK             UERT0830
      IF (IEQDF.EQ.0) WRITE(IOUNIT,45) IER,NAMUPK                       UERT0840
      GO TO 30                                                          UERT0850
   15 CONTINUE                                                          UERT0860
C                                 CHECK FOR UERSET CALL                 UERT0870
      DO 20 I=1,6                                                       UERT0880
         IF (NAMUPK(I).NE.NAMSET(I)) GO TO 25                           UERT0890
   20 CONTINUE                                                          UERT0900
      LEVOLD = LEVEL                                                    UERT0910
      LEVEL = IER                                                       UERT0920
      IER = LEVOLD                                                      UERT0930
      IF (LEVEL.LT.0) LEVEL = 4                                         UERT0940
      IF (LEVEL.GT.4) LEVEL = 4                                         UERT0950
      GO TO 30                                                          UERT0960
   25 CONTINUE                                                          UERT0970
      IF (LEVEL.LT.4) GO TO 30                                          UERT0980
C                                 PRINT NON-DEFINED MESSAGE             UERT0990
      IF (IEQDF.EQ.1) WRITE(IOUNIT,50) IER,NAMEQ,IEQ,NAMUPK             UERT1000
      IF (IEQDF.EQ.0) WRITE(IOUNIT,50) IER,NAMUPK                       UERT1010
   30 IEQDF = 0                                                         UERT1020
      RETURN                                                            UERT1030
   35 FORMAT(19H *** TERMINAL ERROR,10X,7H(IER = ,I3,                   UERT1040
     1       20H) FROM IMSL ROUTINE ,6A1,A1,6A1)                        UERT1050
   40 FORMAT(27H *** WARNING WITH FIX ERROR,2X,7H(IER = ,I3,            UERT1060
     1       20H) FROM IMSL ROUTINE ,6A1,A1,6A1)                        UERT1070
   45 FORMAT(18H *** WARNING ERROR,11X,7H(IER = ,I3,                    UERT1080
     1       20H) FROM IMSL ROUTINE ,6A1,A1,6A1)                        UERT1090
   50 FORMAT(20H *** UNDEFINED ERROR,9X,7H(IER = ,I5,                   UERT1100
     1       20H) FROM IMSL ROUTINE ,6A1,A1,6A1)                        UERT1110
C                                                                       UERT1120
C                                 SAVE P FOR P = R CASE                 UERT1130
C                                   P IS THE PAGE NAMUPK                UERT1140
C                                   R IS THE ROUTINE NAMUPK             UERT1150
   55 IEQDF = 1                                                         UERT1160
      DO 60 I=1,6                                                       UERT1170
   60 NAMEQ(I) = NAMUPK(I)                                              UERT1180
   65 RETURN                                                            UERT1190
      END                                                               UERT1200
```

```
C   IMSL ROUTINE NAME   - UGETIO
C
C-----------------------------------------------------------------------
C
C   COMPUTER            - IBM/SINGLE
C
C   LATEST REVISION     - JUNE 1, 1981
C
C   PURPOSE             - TO RETRIEVE CURRENT VALUES AND TO SET NEW
C                           VALUES FOR INPUT AND OUTPUT UNIT
C                           IDENTIFIERS.
C
C   USAGE               - CALL UGETIO(IOPT,NIN,NOUT)
C
C   ARGUMENTS    IOPT   - OPTION PARAMETER. (INPUT)
C                           IF IOPT=1, THE CURRENT INPUT AND OUTPUT
C                           UNIT IDENTIFIER VALUES ARE RETURNED IN NIN
C                           AND NOUT, RESPECTIVELY.
C                           IF IOPT=2, THE INTERNAL VALUE OF NIN IS
C                           RESET FOR SUBSEQUENT USE.
C                           IF IOPT=3, THE INTERNAL VALUE OF NOUT IS
C                           RESET FOR SUBSEQUENT USE.
C                NIN    - INPUT UNIT IDENTIFIER.
C                           OUTPUT IF IOPT=1, INPUT IF IOPT=2.
C                NOUT   - OUTPUT UNIT IDENTIFIER.
C                           OUTPUT IF IOPT=1, INPUT IF IOPT=3.
C
C   PRECISION/HARDWARE  - SINGLE/ALL
C
C   REQD. IMSL ROUTINES - NONE REQUIRED
C
C   NOTATION            - INFORMATION ON SPECIAL NOTATION AND
C                           CONVENTIONS IS AVAILABLE IN THE MANUAL
C                           INTRODUCTION OR THROUGH IMSL ROUTINE UHELP
C
C   REMARKS      EACH IMSL ROUTINE THAT PERFORMS INPUT AND/OR OUTPUT
C                OPERATIONS CALLS UGETIO TO OBTAIN THE CURRENT U[illegible]
C                IDENTIFIER VALUES. IF UGETIO IS CALLED WITH IOP[illegible] OR
C                IOPT=3, NEW UNIT IDENTIFIER VALUES ARE ESTABLISH[illegible].
C                SUBSEQUENT INPUT/OUTPUT IS PERFORMED ON THE NEW UNITS.
C
C   COPYRIGHT           - 1978 BY IMSL, INC. ALL RIGHTS RESERVED.
C
C   WARRANTY            - IMSL WARRANTS ONLY THAT IMSL TESTING HAS BEEN
C                           APPLIED TO THIS CODE. NO OTHER WARRANTY,
C                           EXPRESSED OR IMPLIED, IS APPLICABLE.
C
C-----------------------------------------------------------------------
C
      SUBROUTINE UGETIO(IOPT,NIN,NOUT)
C                                  SPECIFICATIONS FOR ARGUMENTS
      INTEGER            IOPT,NIN,NOUT
C                                  SPECIFICATIONS FOR LOCAL VARIABLES
      INTEGER            NIND,NOUTD
      DATA               NIND/5/,NOUTD/6/
C                                  FIRST EXECUTABLE STATEMENT
      IF (IOPT.EQ.3) GO TO 10
      IF (IOPT.EQ.2) GO TO 5
      IF (IOPT.NE.1) GO TO 9005
      NIN = NIND
      NOUT = NOUTD
      GO TO 9005
```

```
    5 NIND = NIN                                                    UGET0630
      GO TO 9005                                                    UGET0640
   10 NOUTD = NOUT                                                  UGET0650
 9005 RETURN                                                        UGET0660
      END                                                           UGET0670
```

```
C   IMSL ROUTINE NAME   - USPKD
C
C-----------------------------------------------------------------------
C
C   COMPUTER            - IBM/SINGLE
C
C   LATEST REVISION     - NOVEMBER 1, 1984
C
C   PURPOSE             - NUCLEUS CALLED BY IMSL ROUTINES THAT HAVE
C                           CHARACTER STRING ARGUMENTS
C
C   USAGE               - CALL USPKD  (PACKED,NCHARS,UNPAKD,NCHMTB)
C
C   ARGUMENTS    PACKED - CHARACTER STRING TO BE UNPACKED.(INPUT)
C                NCHARS - LENGTH OF PACKED. (INPUT)  SEE REMARKS.
C                UNPAKD - INTEGER ARRAY TO RECEIVE THE UNPACKED
C                         REPRESENTATION OF THE STRING. (OUTPUT)
C              . NCHMTB - NCHARS MINUS TRAILING BLANKS. (OUTPUT)
C
C   PRECISION/HARDWARE  - SINGLE/ALL
C
C   REQD. IMSL ROUTINES - NONE
C
C   REMARKS  1.  USPKD UNPACKS A CHARACTER STRING INTO AN INTEGER ARRAY
C                IN (A1) FORMAT.
C            2.  UP TO 129 CHARACTERS MAY BE USED.  ANY IN EXCESS OF
C                THAT ARE IGNORED.
C
C   COPYRIGHT           - 1984 BY IMSL, INC.  ALL RIGHTS RESERVED.
C
C   WARRANTY            - IMSL WARRANTS ONLY THAT IMSL TESTING HAS BEEN
C                           APPLIED TO THIS CODE.  NO OTHER WARRANTY,
C                           EXPRESSED OR IMPLIED, IS APPLICABLE.
C
C-----------------------------------------------------------------------
      SUBROUTINE USPKD  (PACKED,NCHARS,UNPAKD,NCHMTB)
C                                  SPECIFICATIONS FOR ARGUMENTS
      INTEGER            NC,NCHARS,NCHMTB
C
      LOGICAL*1          UNPAKD(1),PACKED(1),LBYTE,LBLANK
      INTEGER*2          IBYTE,IBLANK
      EQUIVALENCE (LBYTE,IBYTE)
      DATA               LBLANK /1H /
      DATA               IBYTE /1H /
      DATA               IBLANK /1H /
C                                  INITIALIZE NCHMTB
      NCHMTB = 0
C                                  RETURN IF NCHARS IS LE ZERO
      IF(NCHARS.LE.0) RETURN
C                                  SET NC=NUMBER OF CHARS TO BE DECODED
      NC = MIN0 (129,NCHARS)
      NWORDS = NC*4
      J = 1
      DO 110 I = 1,NWORDS,4
      UNPAKD(I) = PACKED(J)
      UNPAKD(I+1) = LBLANK
      UNPAKD(I+2) = LBLANK
      UNPAKD(I+3) = LBLANK
  110 J = J+1
C                                  CHECK UNPAKD ARRAY AND SET NCHMTB
C                                  BASED ON TRAILING BLANKS FOUND
      DO 200 N = 1,NWORDS,4
```

```
      NN = NWORDS - N - 2                                               USPK0630
      LBYTE = UNPAKD(NN)                                                USPK0640
      IF(IBYTE .NE. IBLANK) GO TO 210                                   USPK0650
  200 CONTINUE                                                          USPK0660
      NN = 0                                                            USPK0670
  210 NCHMTB = (NN + 3) / 4                                             USPK0680
      RETURN                                                            USPK0690
      END                                                               USPK0700
```

LIST OF REFERENCES

1. von Braun, W. and Ordway, F., *History of Rocketry and Space Travel*,3d ed., T. Crowell Co., 1974.

2. Sutton, G.P., *Rocket Propulsion Elements*, 4th ed., John Wiley & Sons, 1976.

3. Jahn, R.G., *Physics of Electric Propulsion*, McGraw-Hill, 1968.

4. Sutton, G.P., *Rocket Propulsion Elements*, 5th ed., John Wiley & Sons, 1986.

5. Stuhlinger, E., *Ion Propulsion for Space Flight*, McGraw-Hill, 1964.

6. Myers, R.M., Mantenieks, M.A., and LaPointe, M.R., *MPD Thruster Technology*, NASA-TM-105242, AIAA-91-3568, Sept., 1968.

7. Myers, R.M., *Applied-Field MPD Thruster Geometry Effects*, AIAA-91-2342, June, 1991.

8. Saber, A.J., *"Anode Power in the Quasi-Steady MPD (Magnetoplasmadynamic) Thruster,"* Ph.D. Thesis, Princeton Univ., NJ, 1974.

9. Shih, K.T., Pfender, E., Ibele, W.E., and Eckert, E.R.G., "Experimental Anode Heat-Transfer Studies in a Coaxial Arc Configuration," *AIAA Journal*, V. 6, No. 8, pp.1482-1487, August, 1968.

10. Sanders, N.A., and Pfender, E., "Measurement of the Anode Falls and Anode Heat Transfer in Atmospheric Pressure, High Intensity Arcs," *J. Appl. Phys.*, V. 55, No. 3, pp. 714-722,Feb.,1984.

11. Vainberg, L.I., Lyubimov, G.A., and Smolin, G.G., "High Current Discharge Effects and Anode Damage in an End-Fire Plasma Accelerator," *Sov. Physics, Tech. Phys.*, V. 23, No. 4, pp. 439-443, April,1978.

12. Hugel, H., "Effects of Self-Magnetic Forces on the Anode Mechanism of a High Current Discharge," *IEEE Trans. Plasma Sci.*, V. PS-8, No. 4, pp.437-442, Dec. 1980.

13. Gallimore, A.D., Kelly, A.J., and Jahn, R.G., "Anode Power Deposition in Quasisteady MPD Thrusters,"*J. Propulsion & Pwr.*, V. 8, No. 6, pp. 1224-1231, Dec., 1992.

14. Dolson, R.C., and Biblarz, O., "Analysis of the Voltage Drop Arising from a Collision-dominated Sheath," *J. Appl. Phys.*, V. 47, No. 12, pp. 5280-5287, Dec., 1976.

15. Myers, R.M., *"Energy Deposition in Low Power Coaxial Plasma Thrusters,"* Ph.D. Thesis, Princeton Univ., NJ, June, 1989.

16. Oberth, R.C., and Jahn, R.G., "Anode Phenomena in High-Current Accelerators," *AIAA Journal*, V. 10, No. 1, pp. 86-91, Jan., 1972.

17. Sleziona, P.C., Auweter-Kurtz, M., and Schrade, H.O., "Numerical Codes for Cylindrical MPD Thrusters," IEPC 88-038, *20th Int'l. Elec. Prop. Conf., Garmisch-Partenkirchen, W. Germany*, Oct., 1988.

18. Sleziona, P.C., Auweter-Kurtz, M., and Schrade, H.O., "Numerical Evaluation of MPD Thrusters," *AIAA 90-2602*, July, 1990.

19. LaPointe, M.R., "Numerical Simulation of Self-Field MPD Thrusters," *AIAA-91-2341, NASA-CR-187168*, July, 1991.

20. Subramaniam, V.V., and Lawless, J.L., "Thermal Instabilities of the Anode in a Magnetoplamsadynamic Thruster," *J. Propulsion & Pwr.*, V. 6, No. 2, pp. 221-224, Mar., 1990.

21. Biblarz, O., "Approximate Sheath Solutions for a Planar Plasma Anode," *IEEE Trans. Plasma Sci.*, V. 19, No. 6, pp. 1235-1243, Dec., 1991.

22. Biblarz, O., Dolson, R.G., and Shorb, R.C., "Anode Phenomena a Collision-dominated Plasma," *J. Appl. Phys.*, V. 46, No. 8, pp. 3342-3346, Aug., 1975.

23. Rudolph,L.K. and Pawlik, E.V.,"The MPD Thruster Development Program,"AIAA Technical Paper 79-2050,in *Progress in Aeronautics and Astronautics*, Vol. 79, Amer. Inst of Aer. & Astro., 1981.

24. Cann, G.L., and Marlotte, G.L.,"Hall Current Plasma Accelerator," *AIAA Journal*, V. 2, No. 7, pp. 1234-1241, Jul., 1964.

25. Brophy, J., *Stationary Plasma Thruster Evaluation in Russia, Summary Report*, Jet Propulsion Laboratory (JPL) Publication 92-4, March 15, 1992.

26. Mitchner, M., and Kruger, C.H., *Partially Ionized Gases*, pp. 128-134, Wiley, 1973.

27. Cobine, J.D.,*Gaseous Conductors, Theory and Engineering Applications*, McGraw-Hill, 1941.

28. von Engel, A., *Ionized Gases*, Clarendon Press, 1965.

29. Chen, F.F., *Introduction to Plasma Physics and Controlled Fusion*, 2nd ed., p.10, Plenum,1984.

30. Lin, S., Resler, E., and Kantrowitz, A., "Electrical Conductivity of Highly Ionized Argon Produced by Shock Waves," *J. Appl. Phys.*, V. 26, p. 95, Jan., 1955.

31. Campbell, A., *Plasma Physics and Magnetofluidmechanics*, p. 161, McGraw-Hill, 1963.

32. Nasser, E., *Fundamentals of Gaseous Ionization and Plasma Electronics*, p.412, Wiley, 1971.

33. Ecker, G., "Anode Spot Instability. I. The Homogeneous Short Gap Instability," *IEEE Trans. Plasma Sci.*, V. PS-2, No. 3, Sept., 1974.

34. Biblarz, O. and Riggs, J.F., "Anode Sheath Contributions in Plasma Thrusters," *AIAA 93-2495*, Jun., 1993.

35. Miller, H.C., "Vacuum Arc Anode Phenomena," *IEEE Trans. Plasma Sci.*, V. PS-11, No. 2, June, 1983.

36. Miller, H.C., "Discharge Modes at the Anode of a Vacuum Arc," *IEEE Trans. Plasma Sci.*, V. PS-11, No. 3, pp. 122-127, Sep., 1983.

37. Rich, J.A., Prescott, L.E., and Cobine, J.D., "Anode Phenomena in Metal-Vapor Arcs at High Currents," *J. Appl. Phys.*, V. 12, No. 12, pp.587-601, Feb., 1971.

38. Schuocker, D., "Improved Model for Anode Spot Formation in Vacuum Arcs," *IEEE Trans. Plasma Sci.*, V. PS-7, No. 4, pp. 209-216, Dec., 1979.

39. Gear, C.W., *Numerical Initial Value Problems in Ordinary Differential Equations*, Prentice-Hall, Englewood Cliffs, NJ, 1971.

40. Burnet, H., Vincent., P., and Rocca Serra, J., "Ionization Mechanism in a Nitrogen Glow Discharge," *J. Appl. Phys.*, V. 54, No. 9, pp. 4951-4957, 1983.

41. Phelps, A.P. and Pitchford, L.C., "Anisotropic Scattering Electrons by N_2 and its Effect on Electron Transport," *Phys. Rev. A*, V. 31, pp. 2932-2949, 1985.

42. Myers, R.M., Kelly, A.J.and Jahn, R.G., "Energy Deposition in Low-Power Coaxial Thrusters," *J. Propulsion*, V. 7, No. 5, pp. 732-739, Sep./Oct., 1991.

43. Biblarz, O., "Thermionic Arc Initiation", *1992 IEEE International Conference on Plasma Science, Tampa, FL*, June 1992.

44. Gallimore, A.D., *"Anode Power Deposition in Coaxial MPD Thrusters,"* Ph.D. Thesis, Princeton Univ., NJ, Oct., 1992.

45. Biblarz, O. and Barto, J.L., "Fluid-Dynamic Effects, Including Turbulence, on a High Pressure Discharge". *Gas Flow and Chemical Lasers*, 6th Int. Symposium (S. Rosenwaks, Ed.), pp. 34-39, Springer-Verlag, Berlin, 1987.

46. Park, W. & Choi, D., "Numerical Analysis of MPD Arcs for Plasma Acceleration," *IEEE Trans. Plasma Sci.*, V. PS-15, No. 5, pp. 618-624, Oct., 1987.

47. Schoeck, P. A., Eckert, E.R.G., and Wutzke, S.A., "An Investigation of the Anode Losses in Argon Arcs and their Reduction by Transpiration Cooling," *ARL 62-341, DTIC AD-278570*, April, 1962.

48. Brady, J.E., *General Chemistry, Principles & Structure*, 5th ed., Wiley, 1990, p. 223.

49. Janes, G.S., "Magnetohydrodynamic Propulsion," in *Advanced Propulsion Techniques*, AGARD Proceedings, Aug., 1960, pp. 151-154, Pergamon, 1961.

50. Connolly, D.J., Sovie, R.J., Michels, C.J., and Burkhart, J.A., "Low Environmental Pressure MPD Arc Tests," *AIAA Journal*, V. 6, No. 7, pp. 1271-1276, July, 1968.

51. Subramaniam, V.V., "Fundamental Studies on Erosion in MPD Thrusters," *AFOSR 87-0360*, April, 1992.

52. Hurwics, H. and Rogan, J. E., "High Temperature Thermal Protection Systems", Section 19, , *Handbook of Heat Transfer* (Rohsenow, W.M., and Hartnett, J.P., Eds) McGraw-Hill, 1973.

53. Kuriki, K. and Suzuki, H., "Quasisteady MPD Arcjet with Anode Gas Injection," AIAA 79-2058, *14th International Electric Propulsion Conference, Princeton, NJ*, Oct., 1979.

54. Sutton, G.W. and Sherman A., *Engineering Magnetohydrodynamics*, p. 148, McGraw-Hill, 1965.

55. Choueiri, E.Y., Kelly, A.J., and Jahn, R.G., "Mass Savings Domain of Plasma Propulsion for LEO to GEO Transfer," *J. Spacecraft and Rockets*, V. 30, No. 6, pp. 749-754, Nov./Dec., 1993.

INITIAL DISTRIBUTION LIST

1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 52 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Chairman Department of Aeronautics & Astronautics, Code AA Naval Postgraduate School Monterey, California 93943-5000	1
4.	Professor Oscar Biblarz Department of Aeronautics & Astronautics, Code AA/Bi Naval Postgraduate School Monterey, California 93943-5000	3
5.	Professor Fred Schwirzke Department of Physics, Code PH/Sw Naval Postgraduate School Monterey, California 93943-5000	1
6.	Commander, Space & Naval Warfare Systems Commamd Space Technology Directorate (SPAWAR-40) 2451 Crystal Drive Arlington, Virginia 22245-5200	1
7.	Dr. Roger Myers Lewis Research Center M.S. SPTD-1 Cleveland, Ohio 44135-3191	1
8.	Dr. James S. Sovey SPTD NASA Lewis Research Center 21000 Brookpark Rd. Cleveland, Ohio 44135	1

9. Dr. Tom Pivirotto — 1
Jet Propulsion Laboratory
4800 Oak Grove Dr.
Pasadena, California 91109
M.S.125-224

10. Dr. John Brophy — 1
Jet Propulsion Laboratory
4800 Oak Grove Dr.
Pasadena, California 91109
M.S.125-224

11. Dr. Jay Polk — 1
Jet Propulsion Laboratory
4800 Oak Grove Dr.
Pasadena, California 91109
M.S.125-224

12. Dr. Arnold J. Kelly — 1
Dept. of Mechanical & Aerospace Engineering
Princeton University
Princeton, New Jersey 08544

13. Dr. Mitat A. Birkan — 1
AFOSR NA
110 Duncan Avenue, Suite B115
Bolling AFB, D.C. 20352-0001

14. Dr. Ed Weiler — 1
Code SZB
NASA Headquarters
Washington, D.C. 20546-0001

15. LCDR John Riggs — 5
10215 Centinella Dr.
La Mesa, California 91941